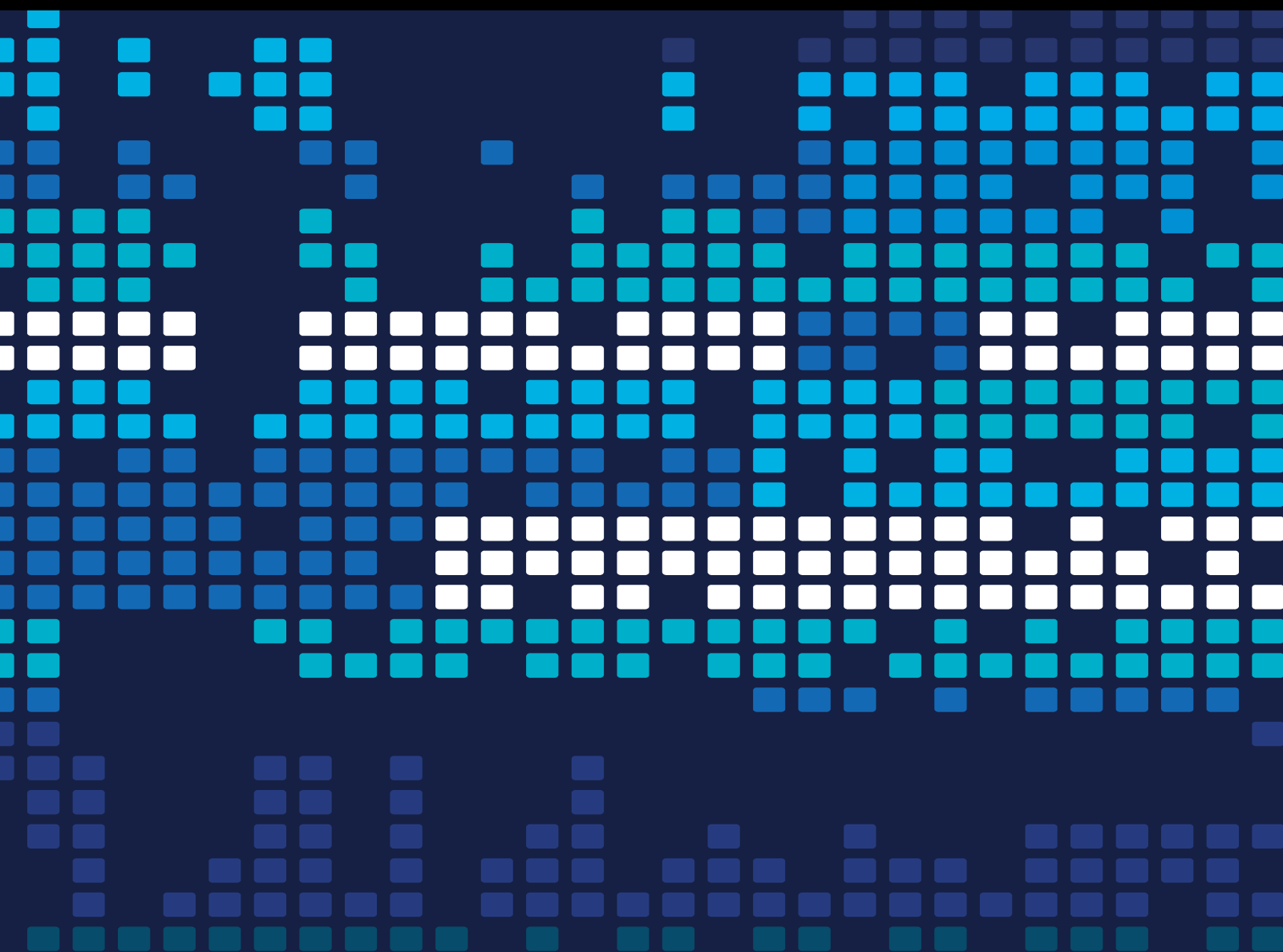


Theory, Algorithms, and Applications for the Multiclass Classification Problem

Lead Guest Editor: KunHong Liu

Guest Editors: Sze-Teng Liong and Wei-Chuen Yau





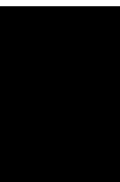
Theory, Algorithms, and Applications for the Multiclass Classification Problem

Scientific Programming

**Theory, Algorithms, and Applications
for the Multiclass Classification
Problem**

Lead Guest Editor: KunHong Liu


Guest Editors: Sze-Teng Liong and Wei-Chuen Yau



Copyright © 2021 Hindawi Limited. All rights reserved.

This is a special issue published in "Scientific Programming." All articles are open access articles distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Chief Editor



Emiliano Tramontana , Italy

Academic Editors





Marco Aldinucci , Italy
Daniela Briola, Italy
Debo Cheng , Australia
Ferruccio Damiani , Italy
Sergio Di Martino , Italy
Sheng Du , China
Basilio B. Fragueta , Spain
Jianping Gou , China
Jiwei Huang , China
Sadiq Hussain , India
Shujuan Jiang , China
Oscar Karnalim, Indonesia
José E. Labra, Spain
Maurizio Leotta , Italy
Zhihan Liu , China
Piotr Luszczek, USA
Tomàs Margalef , Spain
Cristian Mateos , Argentina
Zahid Mehmood , Pakistan
Roberto Natella , Italy
Diego Oliva, Mexico
Antonio J. Peña , Spain
Danilo Pianini , Italy
Jiangbo Qian , China
David Ruano-Ordás , Spain
Željko Stević , Bosnia and Herzegovina
Kangkang Sun , China
Zhiri Tang , Hong Kong
Autilia Vitiello , Italy
Pengwei Wang , China
Jan Weglarz, Poland
Hong Wenxing , China
Dongpo Xu , China
Tolga Zaman, Turkey

Contents







Training Method and Device of Chemical Industry Chinese Language Model Based on Knowledge Distillation

Wen-Ting Li, Shang-Bing Gao , Jun-Qiang Zhang , and Shu-Xing Guo
Research Article (9 pages), Article ID 5753693, Volume 2021 (2021)

Multitask Learning for Aspect-Based Sentiment Classification

Chunhua Yao, Xinyu Song , Xuelei Zhang , Weicheng Zhao , and Ao Feng 
Research Article (9 pages), Article ID 2055555, Volume 2021 (2021)



Sentence Classification Using N-Grams in Urdu Language Text

Malik Daler Ali Awan , Sikandar Ali , Ali Samad , Nadeem Iqbal , Malik Muhammad Saad Missen , and Niamat Ullah 
Research Article (11 pages), Article ID 1296076, Volume 2021 (2021)

The Model of Makerspace Development Element and Performance Analysis Based on NVivo Classification

Yingyan Wang and Rui Zeng 
Research Article (11 pages), Article ID 7123961, Volume 2021 (2021)


DAuGAN: An Approach for Augmenting Time Series Imbalanced Datasets via Latent Space Sampling Using Adversarial Techniques

Andrei Bratu  and Gabriela Czibula 
Research Article (13 pages), Article ID 7877590, Volume 2021 (2021)



A Clustering-based Method for Business Hall Efficiency Analysis

Tianlin Huang  and Ning Wang 
Research Article (12 pages), Article ID 7622576, Volume 2021 (2021)




Quantitative Analysis and Prediction of Global Terrorist Attacks Based on Machine Learning

Xiaohui Pan 
Research Article (15 pages), Article ID 7890923, Volume 2021 (2021)

Grade Prediction in Blended Learning Using Multisource Data

Ling-qing Chen , Mei-ting Wu, Li-fang Pan , and Ru-bin Zheng
Research Article (15 pages), Article ID 4513610, Volume 2021 (2021)

Predicting Learning Behavior Using Log Data in Blended Teaching

Shu-Tong Xie , Zong-Bao He, Qiong Chen , Rong-Xin Chen , Qing-Zhao Kong, and Cun-Ying Song
Research Article (14 pages), Article ID 4327896, Volume 2021 (2021)

The Investigation of Different Loss Functions with Capsule Networks for Speech Emotion Recognition




Anfernee Joan B. Ng  and Kun-Hong Liu 
Research Article (12 pages), Article ID 9916915, Volume 2021 (2021)

How Many Bedrooms Do You Need? A Real-Estate Recommender System from Architectural Floor Plan Images

Y.S. Gan , Shih-Yuan Wang, Chieh-En Huang, Yi-Chen Hsieh, Hsiang-Yu Wang, Wen-Hung Lin, Shing-Nam Chong, and Sze-Teng Liong 



Research Article (15 pages), Article ID 9914557, Volume 2021 (2021)

Improved AND/OR Tree Search Algorithm in Analysis of Stochastic and Time-Dependent Shortest Path Problem

Zhi-ying Xie , Yuan-Rong He , Yuan-tong Jiang, and Chih-Cheng Chen 

Review Article (19 pages), Article ID 9922466, Volume 2021 (2021)

Learning Behavior Analysis Using Clustering and Evolutionary Error Correcting Output Code Algorithms in Small Private Online Courses

Shu-tong Xie , Qiong Chen, Kun-hong Liu , Qing-zhao Kong, and Xiu-juan Cao

Research Article (11 pages), Article ID 9977977, Volume 2021 (2021)

Research Article

Training Method and Device of Chemical Industry Chinese Language Model Based on Knowledge Distillation

Wen-Ting Li, Shang-Bing Gao , Jun-Qiang Zhang , and Shu-Xing Guo

Faculty of Computer and Software Engineering, Huaiyin Institute of Technology, Huaian 223003, China

Correspondence should be addressed to Shang-Bing Gao; gaoshangbing@hyit.edu.cn and Jun-Qiang Zhang; zhangjq0906@hyit.edu.cn

Received 16 August 2021; Revised 29 November 2021; Accepted 1 December 2021; Published 13 December 2021

Academic Editor: Wei-Chuen Yau

Copyright © 2021 Wen-Ting Li et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Recent advances in pretraining language models have obtained state-of-the-art results in various natural language processing tasks. However, these huge pretraining language models are difficult to be used in practical applications, such as mobile devices and embedded devices. Moreover, there is no pretraining language model for the chemical industry. In this work, we propose a method to pretrain a smaller language representation model of the chemical industry domain. First, a huge number of chemical industry texts are used as pretraining corpus, and nontraditional knowledge distillation technology is used to build a simplified model to learn the knowledge in the BERT model. By learning the embedded layer, the middle layer, and the prediction layer at different stages, the simplified model not only learns the probability distribution of the prediction layer but also learns the embedded layer and the middle layer at the same time, to acquire the learning ability of BERT model. Finally, it is applied to the downstream tasks. Experiments show that, compared with the current BERT model distillation method, our method makes full use of the rich feature knowledge in the middle layer of the teacher model while building a student model based on the BiLSTM architecture, which effectively solves the problem that the traditional student model based on the transformer architecture is too large and improves the accuracy of the language model in the chemical domain.

1. Introduction

The past years have seen several major breakthroughs studies that are made in pretrained language models (PLMs) (BERT [1], XLNet [2], RoBERTa [3], SpanBERT [4], and ALBERT [5]). While the learning ability of the current PLMs has been improved a lot significantly, they often have hundreds of millions of parameters, and high computational power should be required. So, it leads to the current PLMs being difficult to apply to solve problems in real life. According to the current research on PLMs, training a large and complex language model still brings great performance on many tasks.

The trend toward bigger models has become inevitable but caused some social concerns. Among them, the most typical one is the BERT model which caused a sensation in the whole NLP world at that time. It had 300 million parameters. The BERT base model is trained on 4 cloud TPUs

(16 TPUs in total). BERT large trains on 16 cloud TPUs (64 TPU chips in total). Each pretraining lasts for 4 days. It follows that the training of the BERT has a high requirement for calculation force and memory. While these PLMs applications are used in real-time operations on devices, which may bring better services, the growing computational and memory requirements of these models may hamper wide adoption.

Many researches showed that the domain-specific pretraining language model can perform better in domain tasks. A large number of corpus and reasonable model structures can make the model better improve its learning ability [6]. The chemical industry belongs to the basic economy of our country and is one of the pillar industries of our country. At the same time, the chemical industry plays an important role in China's economic growth. On March 11, 2019, the International Council of Chemical Associations (ICCA) released a report on the analysis of the contribution of the

chemical industry to the global economy. According to the report, the chemical industry is involved in almost all production industries, and its contribution to the global GDP is estimated to be 5.7 trillion US dollars (7% of the global GDP) through direct, indirect, and induced impacts. Therefore, it is very important to create a pretraining language model in the chemical industry, to solve the text problems in the chemical industry more efficiently.

To create a language model in the chemical field, we have to compress the large model and use a large amount of corpus from the chemical field for training. In the compression, we choose the technology of knowledge distillation, which is different from the previous technology of knowledge distillation [7]. It is not only to learn from the probability distribution of the final output of the teacher model but also to learn in the embedded layer, the middle layer, and the prediction layer. Based on the framework, we propose a training method and device of the Chemical Industry Chinese Language Model based on knowledge distillation.

The main contributions of this work are as follows. (1) Traditional knowledge distillation methods on BERT models often failed to fully learn the representational capabilities of each layer of the teacher model, or to learn these, student models based on transformer architecture still needed to be used, and these student models still had a huge number of parameters. Therefore, we proposed a multilayer BiLSTM architecture for student models to fully learn the representational capabilities of the teacher model, which significantly reduced the number of student model parameters at the expense of only a small portion of performance compared to the former. (2) Nowadays, the pretraining language model is more and more huge, which is difficult to apply in real life. To solve this problem, we have constructed a lightweight multilayer BiLSTM architecture for student models to learn the representational capabilities of the teacher model, and our proposed approach combines a certain level of performance with the lightweight, which is more conducive to be applied to real industry-specific tasks. (3) There are currently no pretrained language models specifically applied to specific chemical industry domains. Based on a large chemical industry corpus, we have constructed a framework of distillation pretrained language models specifically for the chemical industry for later application to specific tasks in the chemical industry.

2. Related Work

2.1. Pretrained Language Models (PLMs). Recently, in the field of natural language processing (NLP), the use of language model pretraining has been improved in several NLP tasks and has been widely concerned. The previous researches on language models mainly include feature-based methods and fine-tuning methods [8, 9]. The details are shown in Table 1.

It can be drawn from Table 1 that the feature-based methods are mainly divided into three types. The first type is the context-independent word representation, mainly including word2vec [10], glove [11], and fastText [12]. The second type is sentence-level representation. For example, continuous

learning for a sentence using Conceptors was proposed by Liu et al. [13], part-of-speech-based long short-term memory network for learning sentence representations was proposed by Zhu et al. [14], and learning sentence representations from explicit discourse relations was proposed by Nie et al. [15]. The third type is the contextualized word representation; the most typical one is the ELMO model. The main feature of the algorithm is that the representation of each word is a function of the entire input sentence. The specific method is to first train the Bidirectional LSTM model with the language model as the target on the large corpus and then use the LSTM to generate the word representation.

The fine-tuning method is to pretrain the language model on a large corpus without monitoring the target and use the labeled data in the domain to fit the model for subsequent applications. The BERT model is one of the models that use this method, but at the same time, although this training paradigm makes the model perform well, the consequent increase in the number of parameters and the long training time makes the model difficult to be applied to real business scenarios. To solve this problem, the article proposes an effective solution, and this method is also suitable for the recently proposed XLNET, RoBERTa, SpanBERT, ALBERT, and other models.

2.2. Knowledge Distillation. To effectively solve the problem of model oversize, we focus on model compression technology [16–18], which can make the model more concise and conducive to application in real life.

The traditional understanding is that training a deep network requires a large number of connections (weights). However, the network training will lead to a high degree of parameter redundancy. The pruning of the network [19–21], reducing the network connections, is a common strategy for model compression. The other direction is weight quantification. In this case, the connection weights are limited to a set of discrete values, with fewer bits representing the weights. However, most of these pruning and quantification techniques [22, 23] are performed on convolutional networks. Only some jobs are designed for specific structural information (such as deep language models [9, 24–26]).

The goal of knowledge distillation is to compress a network with a large number of parameters into a compact and fast working model. This can be achieved by training the compact model to simulate soft inference to a larger model. Mirzadeh et al. [27] proposed a framework based on teacher assistant knowledge distillation. Liu et al. [28] proposed distilling structured knowledge from large networks to compact networks. For the BERT model of distillation compression, Sun et al. [29] first proposed a framework for distillation of the intermediate layers of the BERT model, which takes full advantage of the rich information in the middle hidden layers of the teacher model and encourages the student model to learn and imitate from the teacher model through multilayer distillation. Jiao et al. [30] proposed a two-stage BERT knowledge distillation learning framework that allowed the use of the above distillation in both pretraining and fine-tuning stages, resulting in richer

TABLE 1: The description of previous research on language models.

Feature-based methods	Fine-tuning methods
Context-independent word representation	Pretraining a language model on a large corpus with an unsupervised objective and then fine-tuning the model with in-domain labeled data
Sentence-level representation	
Contextualized word representation	

knowledge learning. Xu et al. [31] proposed a model compression approach that gradually uses small modules to replace modules in BERT, in which only a loss function and a hyperparameter are used to be able to perform model compression without using transformer specific features for compression, which is a general practice. Fu et al. [32] introduced contrastive learning into the construction of distillation loss functions, and the model performance was improved. Feng et al. [33] solved the problem of poor distillation due to lack of data during distillation by means of cross-domain data enhancement. Chen et al. [34] proposed an extraction-then-distillation strategy that reuses the parameters of the teacher model. This strategy can be used for student models of any size, making the student model primed with certain knowledge before the distillation process begins, speeding up convergence, and improving task agnostic distillation efficiency. We studied the problem of compression of linguistic models on a large scale and proposed a training method and device of the Chemical Industry Chinese Language Model based on knowledge distillation to effectively transfer the knowledge of the teacher to the model of the student.

3. Method

In this section, we propose a training method and device of the Chemical Industry Chinese Language Model based on knowledge distillation. The algorithmic procedure flow chart of the proposed framework is shown in Figure 1.

The proposed algorithmic framework consists of a teacher model, a student model, and, most critically, a loss function to connect the two models. First, the framework trained the teacher model, and the raw corpus text was input to the teacher model for training to obtain the trained teacher model weights. Then, the framework started to perform knowledge distillation by distilling the word embedding layer loss, the intermediate layer loss, and the prediction layer loss of the teacher model, respectively, so that the three layers of loss are distilled into the corresponding three-layer BiLSTM model of the student model. Finally, the knowledge distillation ends, and a student model that has learned the performance of the teacher model is obtained. The detailed program algorithm is shown in Algorithm 1.

3.1. Teacher Model. In the teacher model BERT, the original text corpus set T after special processing is first to read and store in T' after processing by line segmentation. The specific

storage format is $T' = \{d_0, d_1, \dots, d_i, \dots\}$, where d_i is the i -th article. d_i stores the collection of all the sentences from article i . $d_i = \{l_0, l_1, \dots, l_j, \dots\}$, where l_j is the j -th sentence in d_i , and $l_j = \{t_0, t_1, \dots, t_k, \dots\}$, where t_k is the k -th token in l_j . Next, the order of articles was scrambled, `dupe_factor` = 10, and then, a random mask was carried out, and $10 * \text{len}(d_i)$ bar samples were generated for each article. While the sampled sentence length exceeds the set maximum sentence length `Lmax` value, the next sentence prediction task in BERT is deleted from the beginning or at the end of one long sentence at random.

Each token in each sentence in T' was sent into BERT's token Embedding Segment Embeddings and Position Embeddings, respectively, and the vector encoding V_1 , the sentence encoding V_2 , and the Position encoding V_3 were obtained, respectively. The vector V_B is obtained by adding the output of the same three dimensions.

BERT in 12 layers of the transformer is cut into 6 layers of the transformer and then will get V_B that is input into the double transformer; BERT and BERT to teacher model covered the token of the probability distribution of m_t and really covered the token vector said m_s loss calculated according to the following formula, where L_t is the random mask task loss function, and then, we carry on the gradient descent optimization model for teachers.

$$L_t(m^s, m^t) = -\text{soft max}(m^t) \cdot \log -\text{soft max}(m^s). \quad (1)$$

3.2. Student Model. In the multilayer neural network model of the student model, first, T is the original text corpus set in the pretreatment of the same as the model of teachers and embedding operation, but the word vector dimensions are half the word vector dimension BERT model, the text of the pretreatment and data input to the multilayer neural network model, the model for the length of the three layers of two-way memory network, in the process of training the student model, and student model by studying the teacher model embedded in the layer, hidden layer, and middle prediction for correction of the model. The network structure of the student model is shown in Figure 2.

In the embedded layer, the specific formula for the loss calculation of the vector output of the embedded layer of BERT and the multilayer neural network of the teacher model and the student model is as follows:

$$L_{\text{emb}}(s_e, t_e) = \text{MSE}(s_e W_e, t_e), \quad (2)$$

where MSE is the Mean Square Error, and matrix $s_e \in \mathbb{R}^{l \times d'}$ and $t_e \in \mathbb{R}^{l \times d}$ embedded, respectively, the student model and

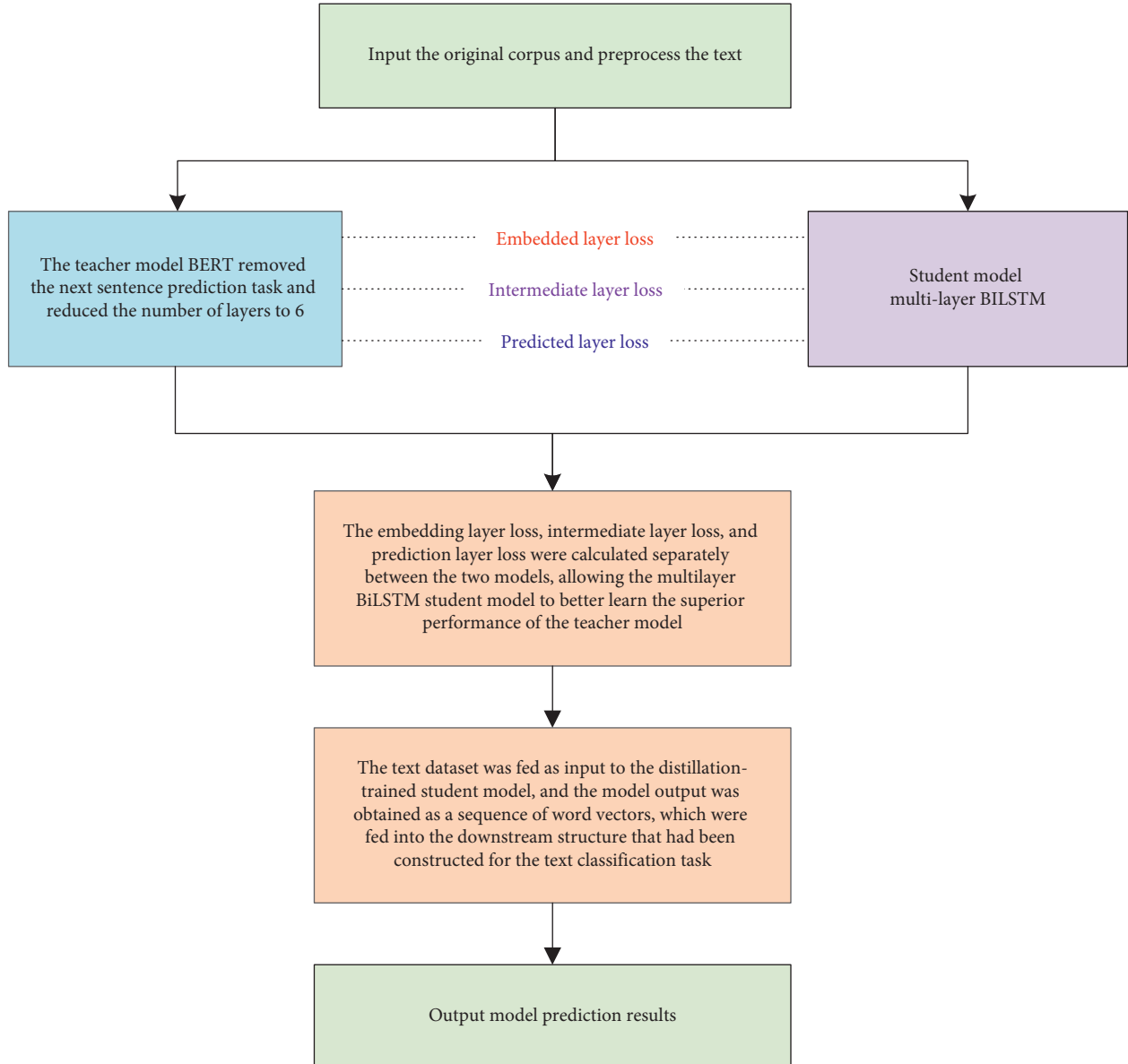


FIGURE 1: Framework program algorithm flow chart.

Input: Training data x , the trained teacher model Bert and its corresponding label

#Initialization

Randomly initialize student parameters θ

#Starting model distillation

While not converge do

For batch data set of x

Extracting word embedding layer loss, intermediate layer loss, and prediction layer loss **from** the teacher model

$t_e, t_h, t_p = \text{Bert}(x)$

Student models begin to distill to learn teacher model knowledge

Minimizing loss function L_{total}

Update parameters θ according to gradients

End for

End while

$$L_{\text{emb}}(s_e, t_e) = \text{MSE}(s_e W_e, t_e),$$

$$L_{\text{hid}}(s_h, t_h) = \text{MSE}(s_h W_h, t_h),$$

$$L_{\text{pre}}(s_p, t_p) = -\text{softmax}(t_p) \cdot \log \text{softmax}(s_p / \text{Tem}),$$

$$L_{\text{total}} = \lambda_e L_{\text{emb}}(e, t_e) + \lambda_{\text{hid}} L_{\text{ht}} + \lambda_{\text{pre}} L_{\text{pre}}(s_p, t_p),$$

$$L_{\text{ht}} = \sum_{h=1}^{h=3} L_{\text{hid}}(s_h, t_{2h-1}),$$

ALGORITHM 1: Continued.

```

# Input  $x$  into the student model BiLSTM to obtain the output as a sequence of word vectors and feed it into the constructed network
for the downstream text classification task
While not converge do
  For batch data set of  $x$ 
     $y = \text{Model}_{\text{classification}}(x)$ ,
    Returns the maximum value in the corresponding dimension, which is the prediction of the classification result
   $y_{\text{pred}} = \arg \max(y)$ 
  End for
End while
Output: prediction results
Remark:  $\theta$  is hyperparameters,  $h$  represents the layer in the student model
    
```

ALGORITHM 1: Distillation to train student models and application of trained student models to downstream text classification tasks.

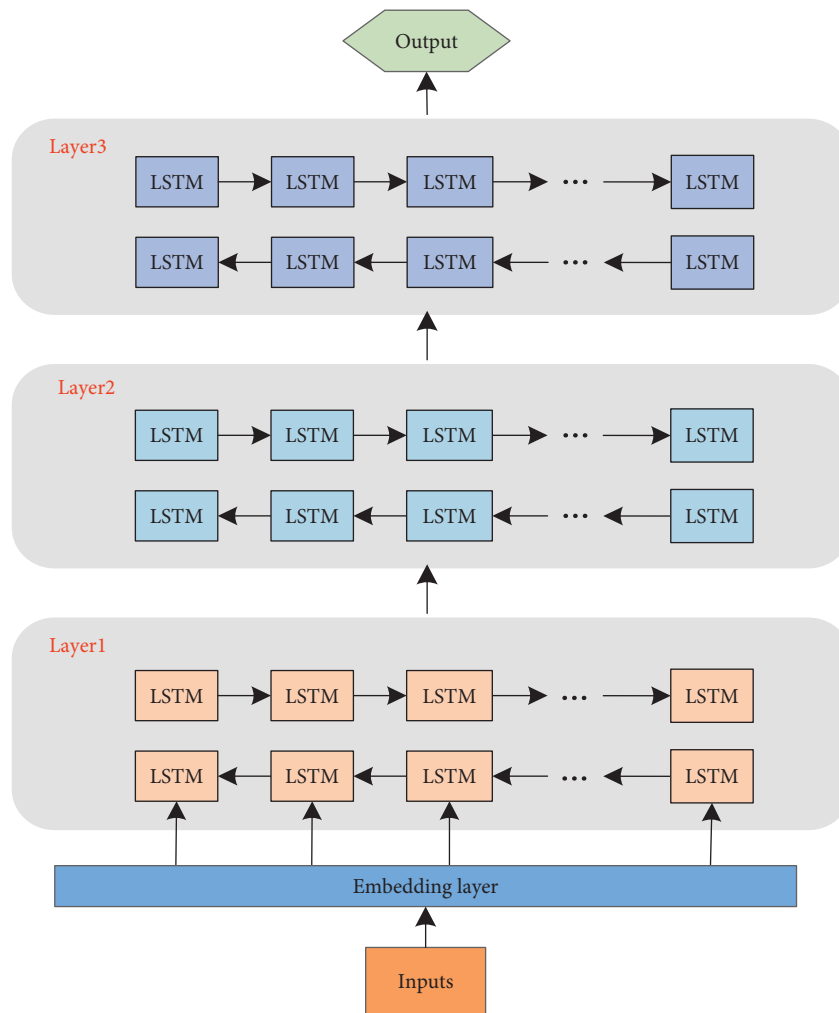


FIGURE 2: Network structure diagram of the student model.

the teacher said. $L = 128$ represents the text length entered by the model, $d = 768$ represents the hidden layer size of the teacher model, and $d' = 200$ represents the hidden layer size of the student model. In the invention, they have the same shape as the hidden state matrix. Matrix $W_e \in \mathbb{R}^{d' \times d}$ is a study of a linear transformation; it will be the student model said embedded into the same teacher's model of state space.

In the middle hidden layer, the output of each hidden layer in the multilayer neural network of the student model and the output of the hidden layer in the transformer corresponding to the teacher model BERT are calculated with MSE mean square error, and the specific formula is shown as follows.

$$L_{\text{hid}}(s_h, t_h) = \text{MSE}(s_h W_h, t_{h'}), \quad (3)$$

where the matrix $s_h \in \mathbb{R}^{1 \times d'}$ and $t_{h'} \in \mathbb{R}^{1 \times d}$, respectively, are students and teachers' network output of hidden layers; matrix $W_h \in \mathbb{R}^{d' \times d}$ linear transformation is learning; it will be the student model of hidden state to transform the same teacher model state space.

In the prediction layer, the probability distribution of the output of BERT's softmax layer of the teacher model and the probability distribution of the output of the softmax layer of the multilayer neural network of the student network are calculated as cross-entropy.

$$L_{\text{pre}}(s_p, t_p) = -\text{soft max}(t_p) \cdot \log_{\text{Tem}} \left(\frac{s_p}{\text{Tem}} \right), \quad (4)$$

where s_p and t_p are, respectively, the logits output predicted by the student model and the teacher model (the input of the upper layer of softmax). $\log_{\text{soft max}}$ is a logarithmic likelihood; $\text{Tem} = 1$ is the temperature value.

By using the above three distillation objectives, the distillation losses of the corresponding layers of the teacher model and the student model can be unified.

$$L_{\text{total}} = \lambda_e L_{\text{emb}}(s_e, t_e) + \lambda_{\text{hid}} L_{\text{ht}} + \lambda_{\text{pre}} L_{\text{pre}}(s_p, t_p), \quad (5)$$

$$L_{\text{ht}} = \sum_{h=1}^{h=3} L_{\text{hid}}(s_h, t_{2h-1}),$$

where L_{ht} represents the loss formula of the total middle hidden layer and s_h and t_{2h-1} , respectively, represent the hidden layer of the h layer of the student model and the output of the second $2h-1$ layer of the corresponding teacher model. $\lambda_e = 1$, $\lambda_{\text{hid}} = 4$, and $\lambda_{\text{pre}} = 3$, respectively, represent the importance of different layers. The specific algorithm structure is shown in Figure 3.

4. Experiments

In this section, we evaluated the performance of this model under the comparison of different experimental models.

The experimental configuration is an AI server configured with $2 \times$ Intel Xeon 6148, 512 g memory, $4 \times$ 1.9 t SSD hard disk, raid card, $2 \times$ ten thousand network card, $8 \times$ Tesla V100 card, $2 \times$ double port 100 Gbps HCA card, 3000 W 1 + 1 redundant server power. And the framework selected for the

experiment was Tensorflow1.12.0. The data set of the Chinese chemical industry in this experiment is from recruitment information of recruitment websites such as Yingcai, the original corpus dataset with a data size of 1,976,522.

4.1. Model Setup. Due to limited equipment, we only retain transformers with 6 layers in the BERT base model, and according to previous studies, we also abandon the next_sentence task to carry out implementation research. The research data shows that the BERT model with 6 layers still has good performance.

We created a multilayer BiLSTM neural network with a hidden size of 200 as a student model. For the BERT model after deletion as a teacher model, with the number of layers being 6, the size of hidden layers is 768 and the head number is 12. The layer mapping function between the teacher model and the student model is $f(h) = 2h-1$. The student model learns at every two layers in the teacher model.

4.2. Teacher Model. Here, we take the BERT model as our teacher model and do not make any settings on the teacher model. Any large pretraining model based on a transformer can be plugged into this framework.

BERT's model architecture is a multilayer bidirectional transformer encoder. BERT base consists of 6 layers, 12 self-attention heads, and 768-dimensional hidden state representation.

Similarly, for the comparison model settings, we changed the layers of the student model accordingly based on the 6-layer teacher model for fairness.

4.3. Student Model. We compare the following models.

4.3.1. Student Model without Distillation. We consider BiLSTM encoders with word embeddings. The last hidden state of BiLSTM is fed into softmax for classification, and the network parameters are trained by optimizing cross-entropy loss over labeled data. We use a basic tokenizer with this model that lowercases all words and splits by whitespace.

4.3.2. Student Model with Distillation. In this, we distill the aforementioned student with (soft/hard) targets and representations from the teacher. First, we fine-tune the teacher on labeled data and use it to generate the logits and hidden state representations for unlabeled instances. We train the student model end-to-end using cross-entropy loss on labeled instances as well as logit loss and representation loss on the unlabeled data. We test three different learning strategies based on a joint optimization scheme as well as two stagewise ones with gradual unfreezing of the intermediate layers.

To verify the validity of the model proposed in this paper, the improved pretraining model is applied to the text classification task. It is noteworthy that we do not compare this model with the recent MobileBERT [35], since the

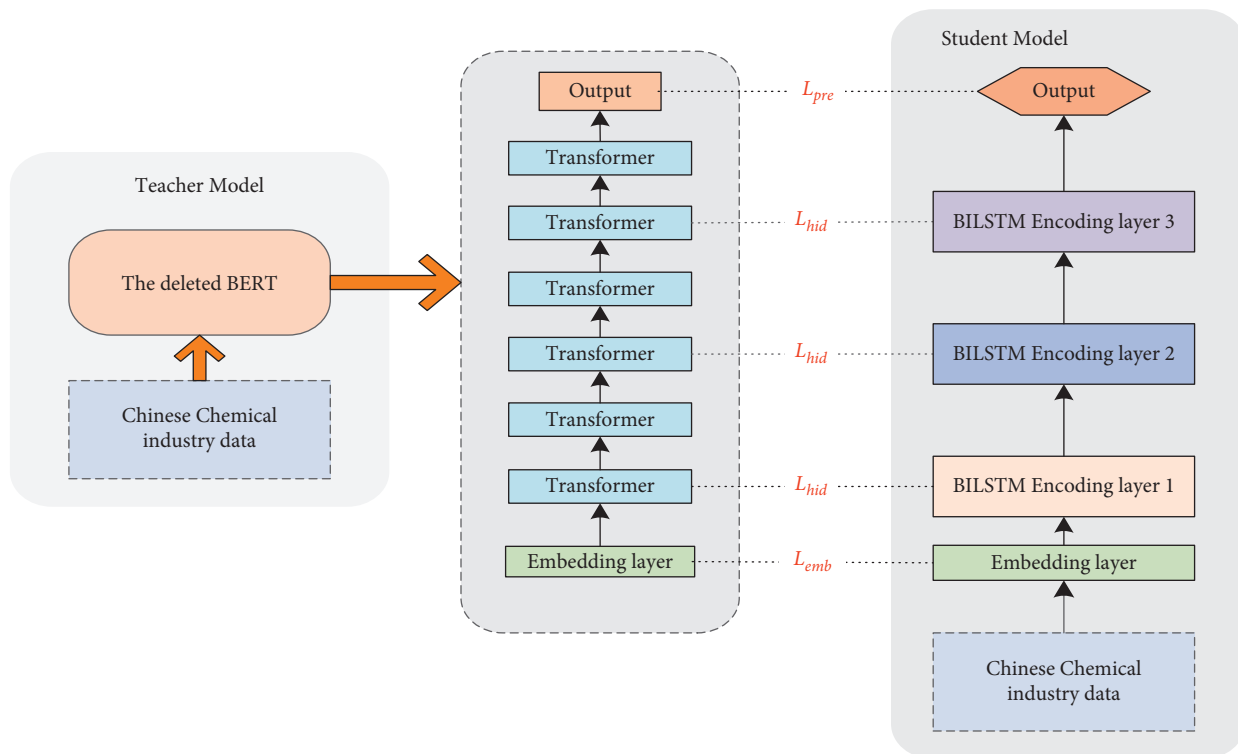


FIGURE 3: The specific algorithm structure.

MobileBERT model employs the transformer block with different architectures.

We created a multilayer BiLSTM neural network with a hidden size of 200 as a student model. For the BERT model after deletion as a teacher model, with the number of layers being 6, the size of hidden layers is 768 and the head number is 12. The layer mapping function between the teacher model and the student model is $f(h) = 2h - 1$. The student model learns at every two layers in the teacher model. The learning weight of each layer is set to $\lambda_e = 1$, $\lambda_{hid} = 4$, and $\lambda_{pre} = 3$, respectively, which performs well for the learning of the student model.

From Table 2, We can find that, compared with the latest method, although our method did not reach the best in the two indicators of accuracy and $F1$ value, it also achieved the third score, which was not a big difference from the latest method, and our proposed method student model was using the BiLSTM architecture with a smaller number of parameters, which made the training speed of the model much faster at the expense of only a small part of the performance. To a certain extent, this also shows that the multilayer BiLSTM architecture can learn the performance of the transformer architecture model very well.

We investigate the effects of distillation objectives on our model learning. Several baselines are proposed including our model learning without the hidden layer distillation (no hidden layer), embedding layer distillation (no embedding layer), and prediction layer distillation (no prediction layer), respectively. The findings are shown in Table 3, which showed that the student model performance could be effectively improved when introducing three layers of loss into

TABLE 2: Distillation performance with BERT base.

Model	Layers	Hidden	Acc (%)	$F1$ (%)
BERT (teacher)	6	768	94.13	92.52
DistillBiLSTM	3	300	91.45	90.21
BERT PKD	3	768	92.87	90.66
DistillBERT [36]	3	768	91.77	89.63
BERT-of-Theseus	3	768	93.43	91.14
BERT-EMD [37]	3	768	93.77	91.34
BiLSTM-KD	3	200	93.13	91.07

TABLE 3: The experimental situation of the model in the absence of different layers.

Distillation details	Layer	Acc (%)	$F1$ (%)
BiLSTM-KD	All layer	93.13	91.07
	No embedding layer	87.44	85.69
	No hidden layer	74.97	71.57
BiLSTM	No prediction layer	84.22	81.26
	—	72.39	70.06

the distillation framework, with the model decreasing more significantly when the middle layer of distillation was removed in the ablation experiment, followed by the prediction layer and word embedding layer. To make the student model fully learn the performance of the teacher model, for the intermediate layer which contained the richest knowledge, this method selected a strategy of extracting the intermediate layers of the teacher model at intervals and distilled them to the student model, so that the student model can better characterize the learning teacher

model performance as a whole. It was also tried to extract only the feature information from the shallow and deeper layers of the teacher model, but the coarse-grained features provided by a single shallow layer or the fine-grained features extracted from the deeper layers could not fully characterize the superior performance of the teacher model, thus resulting in no obvious improvement in the performance of the student model after distillation.

5. Conclusion and Future Work

We proposed a method and device for training Chinese models in the chemical industry based on knowledge distillation. Compared with the traditional distillation model which uses student models based on transformer architecture, this paper constructs a multilayer BiLSTM architecture for student models, so that the superior performance of teacher models can be fully learned using the multilayer structure while further reducing the number of student model participants. Experiments on the text classification task show that the method performed at a somewhat acceptable reduction in performance compared to the baseline model while the number of parameters was significantly reduced, which has important implications for the realistic application of the model to the chemical industry. In future work, we can further consider how to balance the relationship between the number of student model parameters and learning ability, so as to allow the model to be better applied in the industry.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was sponsored by the National Key R&D Program of China (No. 2018YFB1004904), the National Natural Science Foundation of China (61976118), the Key Project of Jiangsu Provincial Department of Education (No. 18KJA520001), Six Talent Peaks Project in Jiangsu Province (XYDXXJS-011), and Jiangsu 333 Engineering Research Funding Project (BRA2016454).

References

- [1] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, "Bert: pre-training of deep bidirectional transformers for language understanding," 2018, <https://arxiv.org/abs/1810.04805>.
- [2] Z. Yang, Z. Dai, Y. Yang, J. G. Carbonell, R. Salakhutdinov, and Q. V. Le, "Xlnet: generalized autoregressive pretraining for language understanding," in *Proceedings of the Advances in Neural Information Processing Systems*, pp. 5753–5763, Vancouver, Canada, December 2019.
- [3] Y. Liu, M. Ott, N. Goyal et al., "Roberta: a robustly optimized bert pretraining approach," 2019, <https://arxiv.org/abs/1907.11692>.
- [4] M. Joshi, D. Chen, Y. Liu, D. S. Weld, L. Zettlemoyer, and O. Levy, "SpanBERT: improving pre-training by representing and predicting spans," *Transactions of the Association for Computational Linguistics*, vol. 8, pp. 64–77, 2020.
- [5] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, "Albert: a lite bert for self-supervised learning of language representations," 2019, <https://arxiv.org/abs/1909.11942>.
- [6] J. Lee, W. Yoon, S. Kim et al., "BioBERT: a pre-trained biomedical language representation model for biomedical text mining," *Bioinformatics*, vol. 36, no. 4, pp. 1234–1240, 2020.
- [7] D. Araci, "Finbert: financial sentiment analysis with pre-trained language models," 2019, <https://arxiv.org/abs/1908.10063>.
- [8] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving Language Understanding by Generative pre-training," 2018.
- [9] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.
- [10] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," *Advances in Neural Information Processing Systems*, vol. 2, pp. 3111–3119, 2013.
- [11] J. Pennington, R. Socher, and C. D. Manning, "Glove: global vectors for word representation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543, Doha, Qatar, 2014.
- [12] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 135–146, 2017.
- [13] T. Liu, L. Ungar, and J. Sedoc, "Continual learning for sentence representations using conceptors," 2019, <https://arxiv.org/abs/1904.09187>.
- [14] W. Zhu, T. Yao, W. Zhang, and B. Wei, "Part-of-speech-based long short-term memory network for learning sentence representations," *IEEE Access*, vol. 7, pp. 51810–51816, 2019.
- [15] A. Nie, E. Bennett, and N. Goodman, "DisSent: learning sentence representations from explicit discourse relations," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4497–4510, Florence, Italy, July 2019.
- [16] J. Gou, B. Yu, S. J. Maybank, and D. Tao, "Knowledge distillation: a survey," 2020, <https://arxiv.org/abs/2006.05525>.
- [17] M. Gupta, V. Varma, S. Damani, and K. N. Narahari, "Compression of deep learning models for NLP," in *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pp. 3507–3508, Bangalore, India, October 2020.
- [18] D. H. Le, V. T. Nhan, and N. Thoai, "Paying more attention to snapshots of iterative pruning: improving model compression via ensemble distillation," 2020, <https://arxiv.org/abs/2006.11487>.
- [19] P. Molchanov, A. Mallya, S. Tyree, Frosio, and Kautz, "Importance estimation for neural network pruning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 11264–11272, Long Beach, CA, USA, June 2019.
- [20] X. Dong and Y. Yang, "Network pruning via transformable architecture search," in *Proceedings of the Neural Information*

- Processing Systems (NeurIPS)*, pp. 760–771, Vancouver, Canada, December 2019.
- [21] D. Blalock, J. J. G. Ortiz, J. Frankle, and J. Guttag, “What is the state of neural network pruning?,” 2020, <https://arxiv.org/abs/2003.03033>.
 - [22] J. Wang, M. Ramajayam, E. Charrault, and N. Stanford, “Quantification of precipitate hardening of twin nucleation and growth in Mg and Mg-5Zn using micro-pillar compression,” *Acta Materialia*, vol. 163, pp. 68–77, 2019.
 - [23] A. Carpentier, J. Eisert, D. Gross, and R. Nickl, “Uncertainty quantification for matrix compressed sensing and quantum tomography problems,” *High Dimensional Probability VIII*, Birkhäuser, Cham, Switzerland, pp. 385–430, 2019.
 - [24] K. Irie, A. Zeyer, R. Schlüter, and H. Ney, “Language modeling with deep transformers,” 2019, <https://arxiv.org/abs/1905.04226>.
 - [25] X. Cheng, “Dual-view distilled BERT for sentence embedding,” 2021, <https://arxiv.org/abs/2104.08675>.
 - [26] P. Liu, X. Wang, L. Wang, W. Ye, X. Xi, and S. Zhang, “Distilling knowledge from BERT into simple fully connected neural networks for efficient vertical retrieval,” in *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pp. 3965–3975, 2021.
 - [27] S. I. Mirzadeh, M. Farajtabar, A. Li, N. Levine, A. Matsukawa, and H. Ghasemzadeh, “Improved knowledge distillation via teacher assistant,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 4, pp. 5191–5198, Vancouver, Canada, February 2020.
 - [28] Y. Liu, C. Shu, J. Wang, and C. Shen, “Structured knowledge distillation for dense prediction,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, p. 1, 2020.
 - [29] S. Sun, Y. Cheng, Z. Gan, and J. Liu, “Patient knowledge distillation for bert model compression,” 2019, <https://arxiv.org/abs/1908.09355>.
 - [30] X. Jiao, Y. Yin, L. Shang et al., “TinyBERT: distilling bert for natural language understanding,” in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pp. 4163–4174, Punta Cana, Dominican Republic, December 2020.
 - [31] C. Xu, W. Zhou, T. Ge, F. Wei, and M. Zhou, “Bert-of-theseus: compressing bert by progressive module replacing,” 2020, <https://arxiv.org/abs/2002.02925>.
 - [32] H. Fu, S. Zhou, Q. Yang et al., “LRC-BERT: latent-representation contrastive knowledge distillation for natural language understanding,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 14, Article ID 12830, Vancouver, Canada, March 2021.
 - [33] L. Feng, M. Qiu, Y. Li, H. Zheng, and Y. Shen, “Learning to augment for data-scarce domain BERT knowledge distillation,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 7422–7430, Vancouver, Canada, February 2021.
 - [34] C. Chen, Y. Yin, L. Shang et al., “Extract then distill: efficient and effective task-agnostic bert distillation,” 2021, <https://arxiv.org/abs/2104.11928>.
 - [35] Z. Sun, H. Yu, X. Song, R. Liu, Y. Yang, and D. Zhou, “Mobilebert: a compact task-agnostic bert for resource-limited devices,” 2020, <https://arxiv.org/abs/2004.02984>.
 - [36] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, “DistilBERT, a distilled version of bert: smaller, faster, cheaper and lighter,” 2019, <https://arxiv.org/abs/1910.01108>.
 - [37] J. Li, X. Liu, H. Zhao, R. Xu, M. Yang, and Y. Jin, “BERT-EMD: many-to-many layer mapping for bert compression with earth mover’s distance,” 2020, <https://arxiv.org/abs/2010.06133>.

Research Article

Multitask Learning for Aspect-Based Sentiment Classification

Chunhua Yao,¹ Xinyu Song ,² Xuelei Zhang ,² Weicheng Zhao ,² and Ao Feng ²

¹The 30th Research Institute of China Electronics Technology Group Corporation, Chengdu, China

²Chengdu University of Information Technology, Chengdu, China

Correspondence should be addressed to Xinyu Song; xinyu.s@foxmail.com

Received 20 July 2021; Revised 8 October 2021; Accepted 8 November 2021; Published 29 November 2021

Academic Editor: Sze-Teng Liong

Copyright © 2021 Chunhua Yao et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Aspect-level sentiment analysis identifies the sentiment polarity of aspect terms in complex sentences, which is useful in a wide range of applications. It is a highly challenging task and attracts the attention of many researchers in the natural language processing field. In order to obtain a better aspect representation, a wide range of existing methods design complex attention mechanisms to establish the connection between entity words and their context. With the limited size of data collections in aspect-level sentiment analysis, mainly because of the high annotation workload, the risk of overfitting is greatly increased. In this paper, we propose a Shared Multitask Learning Network (SMLN), which jointly trains auxiliary tasks that are highly related to aspect-level sentiment analysis. Specifically, we use opinion term extraction due to its high correlation with the main task. Through a custom-designed Cross Interaction Unit (CIU), effective information of the opinion term extraction task is passed to the main task, with performance improvement in both directions. Experimental results on SemEval-2014 and SemEval-2015 datasets demonstrate the competitive performance of SMLN in comparison to baseline methods.

1. Introduction

Sentiment analysis is one of the fundamental tasks in natural language processing and has received an increasing level of attention in recent years. Aspect-level sentiment classification focuses on fine-grained sentiment analysis and is widely applied to automatic processing tasks for online review text. The purpose of this task is to determine the emotional polarity of entities in each aspect of a review piece [1–4], with each entity consisting of one or multiple words. The number of aspect terms in a sentence is arbitrary [5–9], and each aspect may carry a different sentiment polarity. Within the sentence “I love this program, it is superior to windows movie maker” in Figure 1, “program” and “windows movie maker” are two separate aspect terms, but they carry positive and negative emotions, respectively. In the example above, “love” and “superior” are defined as opinion terms. It can be observed that the emotional polarity of aspect words comes from their corresponding opinion words.

Existing algorithms for aspect-level sentiment analysis are mainly divided into feature engineering methods and deep learning models. For the methods based on feature

engineering, the main idea is to design a series of handcraft features, and a traditional classifier is trained to achieve high emotion classification accuracy [5, 10, 11]. This class of methods consumes a lot of manpower, and the vocabulary dependency on individual scenarios makes it difficult to generalize.

Models based on deep neural networks have better potential in solving these issues. Through multilayer neural networks, low-dimensional vectors representing term semantics can be effectively trained without complex feature engineering process. These embedding representations become the input of downstream neural networks, in order to identify the emotional polarity of target words. Satisfactory results have been achieved through various target-dependent sentiment mechanisms [5–9]. These Long Short-Term Memory (LSTM)-based methods take static word vectors such as Word2Vec and GloVe as input and use the feature representation of entity words for sentiment classification. They simply fuse contextual information into the representation of the target word, without considering their semantic correlation. Recent research efforts apply the attention mechanism to consider interaction between the

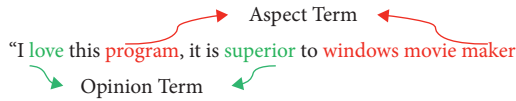


FIGURE 1: An example of consumer review that contains aspect and opinion terms.

aspect words and their context. A variety of complex structures were designed to calculate attention weights between aspect words and their context [12–15]. As most aspect-level datasets do not come in large scale, the risk of overfitting greatly increases with the model size and complexity. As a result, methods based on the attention mechanism tend to make more mistakes when mining deeper features.

In recent years, multitask learning (MTL) becomes an active research area in machine learning, which improves the generalization performance of a task by jointly training other related tasks [16]. Due to the success of MTL [17], there are several NLP models based on neural networks that adopt this mechanism [18–20]. By using shared representations to learn semantically related tasks in parallel, MTL captures the correlation between tasks, improving the generalization ability of the model under certain circumstances. The multitasking architecture of these models contains shared lower layers to train their common features, and the remaining layers are customized to handle different tasks. For aspect-level sentiment classification, [21, 22] have made some attempts in MTL, and their study shows that joint training with document-level sentiment classification can significantly improve performance at the aspect level. Yu and Jiang [23] design an auxiliary task that is highly relevant to sentiment analysis, which predicts whether the input sentence contains positive or negative words.

In this study, we propose a Shared Multitask Learning Network (SMLN), which employs opinion word extraction as an auxiliary task and trains it together with aspect-level sentiment analysis. This task is an upstream step that extracts key opinion terms, and its performance also affects the accuracy of the main task. The pretrained BERT model is used as the underlying structure shared by the two tasks. SMLN introduces a new feature sharing mechanism, Cross Interaction Unit (CIU), to facilitate the information exchange between the main task and the auxiliary task. Specifically, CIU consists of multiple groups of attention mechanisms, integrating the information of the two tasks from different viewing angles. With extensive experiments on the SemEval-2014 and SemEval-2015 datasets, the results indicate that our SMLN model outperforms other baseline methods in terms of classification accuracy. For a fair comparison, some of the baseline methods are built on the same BERT-based representation, so the performance boost is originated from the multitask setting and information sharing mechanism.

Main contributions of the work include the following: (1) a multitask learning method customized for fine-grained sentiment analysis, which utilizes additional opinion word information to improve the learning performance of the current task and reduce the risk of overfitting; (2) a multitask

sharing mechanism to accomplish multiview information transfer between tasks.

2. Related Work

In this section, relevant research on aspect-level sentiment analysis and multitask learning is reviewed, covering both traditional neural networks and more recent BERT-based model.

2.1. Conventional Neural Network. In aspect-level sentiment analysis, it has been agreed among researchers that context words have different influence on the sentiment polarity of multiple targets in the opinion sentence. When a learning system is built for sentiment classification, the most important task is to integrate the relationship between each target and its context. Vo and Zhang [7] divide the original sentence into the target words, the left context, and the right context and use different networks to extract their features. Tang et al. [8] propose a target-dependent LSTM model. In order to represent the characteristics of the aspect word more accurately, they use two LSTMs to encode the previous context and the next context including the target itself.

Previous studies try to establish the connection between the target words and their context, but the interaction information is hard to capture with canonical methods. Wang et al. [12] propose an LSTM network based on the attention mechanism, which is used for aspect-level sentiment classification. When different aspects are involved, this model automatically directs its attention to different parts of the sentence. Li et al. [13] design an end-to-end structure, constantly focusing on an aspect term and its context. Ma et al. [14] believe that entities and corresponding contexts can be reasonable, and calculate attention weights for target and context, respectively. Fan et al. [15] propose a multi-grained attention network. In addition to the attention calculation of the aspect words and the overall context, they also introduce fine-grained attention. The purpose is to describe the influence of an aspect on its context, or context on aspect words in the reverse direction. Tang et al. [24], Chen et al. [25], and Zhu and Qian [26] propose a readable and writable external memory module, which shows the contribution of each word to the final sentiment classification.

2.2. Multitask Learning. All methods mentioned above focus on a single task and obtain acceptable performance in aspect-level sentiment analysis. If related tasks are built on the same text representation and language modeling, it is possible to further improve the learning performance of the main task. In recent years, studies have adopted multitask learning methods to handle fine-grained sentiment analysis. Yu and Jiang [23] design an auxiliary task to determine if an input sentence contains positive or negative sentiment words. This auxiliary task is closely related to the main task of sentiment analysis. They propose to train the sentiment analysis task and the hidden feature representation task together, and the auxiliary task helps in generating more

representative embedding for sentiment analysis sentences. He et al. [21] propose a multitask learning method that jointly trained the document-level and aspect-level sentiment classification tasks. Leveraging the information at the document level, current aspect model’s limitation is alleviated with the introduction of the larger datasets. He et al. [22] employ an information delivery mechanism so that identical implicit expressions can be shared among multiple tasks in an iterative manner. This multitask model can utilize more global knowledge to improve the accuracy of sentiment analysis.

2.3. BERT-Based Networks. Traditional neural networks include structures like LSTM or multilayer CNN as encoders and use word vectors generated by Word2Vec or GloVe. However, the performance of these models is limited by the static nature of word vectors, and the downstream networks cannot break the performance bottleneck imposed by the text representation. In order to solve this problem, recent research has focused on large-scale pretrained attention models like ELMo, GPT, and BERT, and gains have been observed in many text applications with the richer representation. Sun et al. [27] propose four methods of constructing auxiliary sentences with the aspect term, feeding auxiliary sentences together with the original sentence as the input to the BERT model. This method transforms aspect-level sentiment analysis into a sentence pair classification task. Xu et al. [28] learn domain knowledge based on large-scale pretraining with BERT in the same domain as the original dataset. Gao et al. [29] design an intuitive method to use the feature expression of the target word on aspect-level sentiment classification, with slight modification of the BERT model. Zhou et al. [30] use the graph convolutional network (SK-GCN) model of grammar and knowledge to enhance the representation of sentences for given aspects. Song et al. [31] propose a semantics perception and refinement network (SPRN) for sentiment analysis based on aspects. Local semantic features are extracted by multi-channel convolution operation. They use gated networks to enhance aspect and context connections while filtering noise.

3. The Approach

The SMLN architecture is shown in Figure 2. In this section, we will elaborate on the details of the SMLN structure for aspect-level sentiment classification. It starts with the definition of the main task and auxiliary tasks, together with the necessary notations. Then, the BERT-based representation is included as a shared layer. CIU, which is the unit that establishes interaction between the main and auxiliary tasks, is introduced after that.

3.1. Problem Definition and Notations. s_i denotes a sentence from the training dataset, which consists of a sequence of tokens:

$$s_i = [w_1, w_2, \dots, w_i, \dots, w_n]. \quad (1)$$

Sentence s_i includes target words t that need to be annotated with their polarity of sentiment, and opinion words o that carry the corresponding emotion information. A target

$$t = [w_j, w_{j+1}, \dots, w_{j+m-1}], \quad (2)$$

contains m words; the opinion terms

$$o = [w_j, w_{j+1}, \dots, w_{j+k-1}], \quad (3)$$

include k words; and t and o are both subsequences of s . For the main task, aspect-level sentiment classification (ASC), its goal is to determine the emotional polarity of the target word t in the sentence s_i . Available tags include “positive,” “negative,” and “neutral.” For the auxiliary task, opinion terms extraction (OTE), its aim is to extract all the opinion terms appearing in a sentence. For simplicity, we treat OTE as a sequence tagging problem, with the BIO tagging scheme. Specifically, we use three categories of tags: $Y^{\text{OTE}} = \{B, I, O\}$ indicating the beginning and interior of the opinion term, and other words, respectively. For example, for the sentence “The screen is large and crystal clear with amazing colors”, its opinion extract label is shown in Table 1.

3.2. Shared Layer. The input embedding layer maps the original text representation into a high-dimensional vector space. The pretrained BERT model is employed to obtain embedding representation of each word with fine-tuning capability in the original Transformer network. BERT [32] is one of the leading language representation models, which uses a bidirectional Transformer [33] network to pretrain a language model on a large text corpus, and the pretrained representation can be fine-tuned on other tasks. The task-specific BERT design is able to represent either a single sentence or a pair of sentences as a consecutive array of tokens. For a given token, its input representation is constructed by adding up its corresponding token, segment, and position embeddings. For a typical classification task, the first word of the sequence is identified with a unique token [CLS], and a fully connected layer is attached at the [CLS] position of the last encoder layer. The last layer is usually softmax which completes the classification task.

BERT has two parameter intensive settings:

BERT_{base}: The number of Transformer blocks is 12, the hidden layer size is 768, the number of self-attention heads is 12, and the total number of parameters for the pretrained model is 110M.

BERT_{large}: The number of Transformer blocks is 24, the hidden layer size is 1024, the number of self-attention heads is 16, and the total number of parameters for the pretrained model is 340M.

The BERT_{large} model requires considerably more memory than BERT_{base}. As a result, the maximal batch size for BERT_{large} is so small on a single GPU with limited memory that it actually hurts the model accuracy, regardless of the learning rate [32]. Therefore, we use BERT_{base} as our baseline model, with modifications that do not significantly increase the model size.

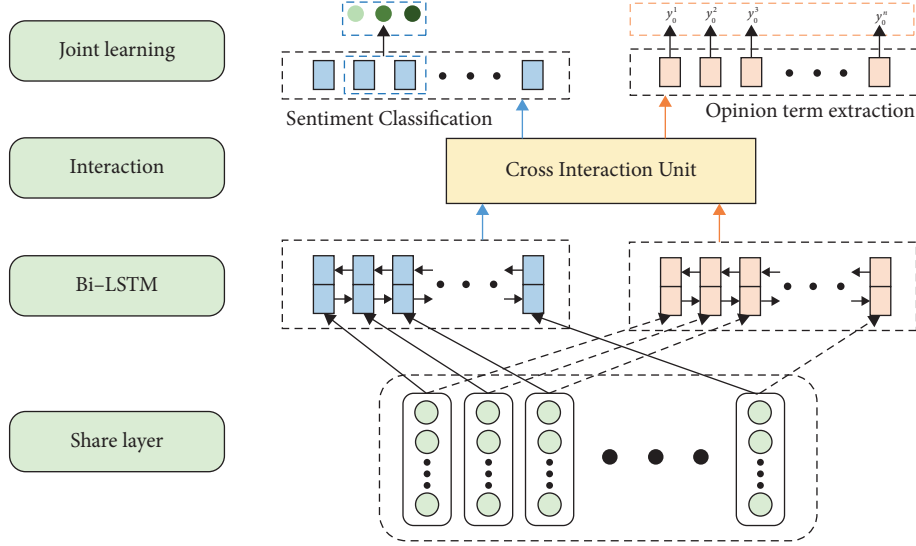


FIGURE 2: Framework of the proposed SMLN.

TABLE 1: An auxiliary task training instance with gold opinion extract labels.

Input	The	Screen	Is	Large	And	Crystal	Clear	With	Amazing	Colors
Label	O	O	O	B	O	B	I	O	B	O

Following annotations in the previous section, a sentence with size n contains a target/aspect that is composed of m terms. BERT uses WordPiece [34] as its tokenizer. After the multilayer bidirectional Transformer network, the word vector matrix S_r of the sentence S is represented by the hidden status of the last layer.

$$S_r = [x_0, x_1, x_2, \dots, x_i, \dots, x_n, x_{n+1}], \quad (4)$$

$S_r \in R^{(n+2) \times d}$, where d is the dimension of hidden state. x_0 is the vector of the sentence classification mark [CLS], and $x_{(n+1)}$ is the word vector of the sentence separator [SEP]. Then, we use two Bi-LSTM networks to decompose S_r . The outputs of the two networks are denoted as S_p and S_o , which focus on ASC and OTE, respectively.

3.3. Cross Interaction Unit. When generating the representation with two independent Bi-LSTMs, the information of ASC and that of OTE, as two individual tasks, are separated from each other. However, the reality is that the two parts are closely related. For example, when ‘love’ appears around an aspect term, its polarity is likely to be positive. The Cross Interaction Unit (CIU) is designed to exchange information between these tasks, mining opinion terms and identifying aspect sentiment polarity in a cooptimization manner. The CIU architecture is shown in Figure 3.

A basic CIU is composed of a pair of attention modules: polarity attention and opinion attention. We define the output $S_p = \{p_0, p_1, \dots, p_n, p_{n+1}\}$ and $S_o = \{o_0, o_1, \dots, o_n, o_{n+1}\}$ of two Bi-LSTM networks to represent the distribution of sentiment feature and opinion feature, respectively, where $p_i \in R^{2d}$ and $o_i \in R^{2d}$ are representations of the i -th token w_i . P and O are input to the emotional attention module, in which

we first calculate the composition vector $\alpha_{ij}^p \in R^K$ between P and O through a tensor operation:

$$\alpha_{ij}^p = f_p(p_i, o_j) = \tanh((p_i)^T G_p o_j), \quad (5)$$

where $G_p \in R^{K \times 2d \times 2d}$ is a 3-dimensional tensor. A tensor operator can be viewed as multiple bilinear terms that are capable of modeling more complicated compositions between two vectors [35]. K is a hyperparameter representing the number of G_p channels. Each channel of G_p is a bilinear term that can extract specific information. A larger number of K represents the complicated intrinsic correlation between sentiment classification features and opinion extraction features. As the value of K increases, more information is extracted together with higher complexity.

After obtaining the composition vectors, the attention score e_i^p for token w_i is calculated as

$$e_i^p = v^p \alpha_{ij}^p. \quad (6)$$

Here, $v^p \in R^K$ can be seen as a weight vector to measure each value of the composition vector. e_{ij}^p is a scalar value that composes a matrix E_p . A higher score for e_{ij}^p indicates that the current sentiment feature of the i -th word captures more information from the opinion expression of the j -th word. Finally, we fuse the sentiment feature P and opinion feature O generated by the original Bi-LSTM as follows:

$$\begin{aligned} S'_p &= S_p + \text{softmax}_r(E_p)S_o, \\ S'_p &= \{p'_0, p'_1, \dots, p'_n, p'_{n+1}\}, \end{aligned} \quad (7)$$

where softmax_r is a row-based softmax function. S'_p represents the final sentiment expression of the sentence. Similarly, we can get the final expression of the opinion

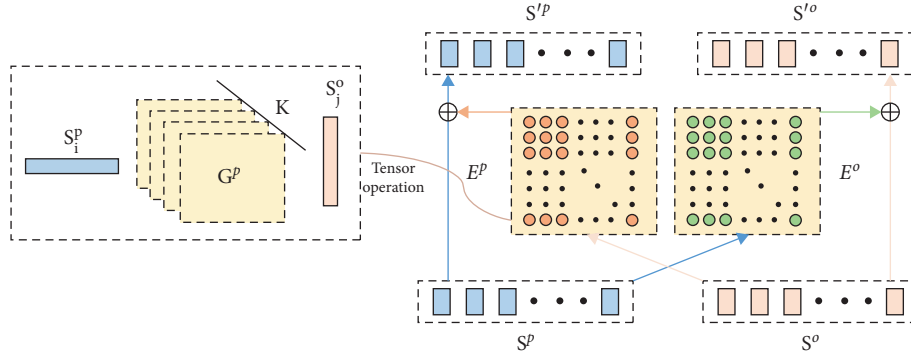


FIGURE 3: The Cross Interaction Unit.

feature S'_0 . With the crossover operations in CIU, the accuracy of sentiment classification and opinion extraction can be improved simultaneously.

The vector of the target word obtained from S'_p is

$$Tr = [p'_i, p'_{i+1}, \dots, p'_{i+m-1}], \quad (8)$$

where $Tr \in R^{m \times 2d}$ and m represents the length of the target word. A max-pooling operation is performed on the target word vector, and the most important features at each position are selected from different words.

$$V = \max\{Tr, \dim = 0\}, V \in R^{1 \times d}. \quad (9)$$

Finally, V is fed into a fully connected layer for classification. For the opinion extraction task, we use a dense layer plus a softmax operation to generate the final opinion tags.

3.4. Joint Learning. Output from the previous step contains representation of the original text for two purposes: one is polarity labeling, and the other is opinion term labeling. These tasks require different forms of output, so it is necessary to apply gradient descent training that better fit their respective application.

For the sentiment classification branch, V represents the polarity characteristics of the target word. It passes through the fully connected softmax layer to obtain probability values representing emotion polarities.

$$p(a) = \text{softmax}(W_p V_a + b_p). \quad (10)$$

After that, we use the standard cross-entropy loss as the cost function:

$$\text{loss}_p = - \sum_{a \in D} \sum_{c=1}^C P_c^g(a) \cdot \log(P_c(a)), \quad (11)$$

where a represents an aspect term appearing in training data D . C represents the number of categories of sentiment classification. $P_c(a)$ is the probability of predicting s as class c from the softmax layer, and $P_c^g(a)$ indicates whether class c is the correct sentiment category, with value 1 or 0.

For the opinion term extraction branch, all possible outputs of the tag sequence are defined as array Y . $\mathcal{Y}_{\text{real}}$ is the true label sequence. From the SMLN, the feature value of

each location is converted into a probability value through softmax, and the formula is as follows:

$$p(\mathcal{Y}_{\text{real}}|X) = \frac{e^{S(X, \mathcal{Y}_{\text{real}})}}{\sum_{y \in Y} e^{S(X, y)}}. \quad (12)$$

The goal of model optimization is to increase the probability of the appearance of the true label and ultimately reduce the value of the loss function. The objective loss function of the opinion word recognition model is defined as follows:

$$\text{loss}_o = -\log(p(y_{\text{real}}|X)) = -S(X, y_{\text{real}}) + \log\left(\sum_{y \in Y} e^{S(X, y)}\right). \quad (13)$$

Losses of the main sentiment classification task and the auxiliary opinion word extraction task are aggregated to form the total loss $\mathcal{F}(\theta)$ of the framework.

$$\mathcal{F}(\theta) = \text{Loss}_p + \text{Loss}_o. \quad (14)$$

4. Experiments

4.1. Datasets. Our framework is evaluated on three benchmark datasets from SemEval-2014 [4] and SemEval-2015 [36]. Statistics of the datasets are shown in Table 2. For simplicity, we use 14Lap, 14Res, and 15Res to denote SemEval-2014 Laptops, SemEval-2014 Restaurants, and SemEval-2015 Restaurants, respectively. There are four emotional labels in the entities in datasets, which are “positive,” “negative,” “natural,” and “conflict.” “Conflict” means that there are more than two emotions in the same entity. Labels on opinion terms are provided by Wang et al. [35, 37].

4.2. Experiment Settings. The pretrained uncased BERT-base model is used for fine-tuning. The number of Bi-LSTM hidden units is set to 300, and the output dimension of Bi-LSTM is 600. The hyperparameter K is set to 5. In the fine-tuning process, the same parameter settings as the BERT model are kept to ensure comparable results to other baseline models. To avoid overfitting, the dropout

TABLE 2: Statistics of the experiment datasets.

Dataset	Positive	Negative	Neutral	Conflict	Total	Opinion
14Lap-Train	987	866	460	45	2358	2504
14Lap-Test	341	128	169	16	654	674
14Res-Train	2164	805	633	91	3693	3484
14Res-Test	728	196	196	14	1134	1008
15Res-Train	902	252	34	—	1315	1210
15Res-Test	319	319	27	—	685	510

probability value is set as 0.5 for the opinion extract. The model is implemented with the PyTorch library and runs on a single Nvidia 2080 Ti GPU. Other hyperparameters are shown in Table 3.

5. Results

5.1. Baseline Approaches. Following the convention of related work, the average accuracy metric is used to measure the overall performance of sentiment classification models. To show the effectiveness of our model, several mainstream models for aspect-based sentiment analysis are used for comparison, including the following:

TD-LSTM [8] uses two LSTM networks to model the correlation between the target word and its context, and it concatenates the last hidden state of the two parts to predict the sentiment polarity of the target.

ATAE-LSTM [12] applies a typical attention-based LSTM structure to capture the key part of the sentence in response to a given aspect.

MemNet [24] is a deep memory network that applies multiple attention layers to capture the importance of each context word and predicts sentiments based on the sentence representation at the top level.

RAM [25] has a multilayer architecture where each layer consists of an attention-based aggregation of word features and a GRU cell to strengthen the expressive power of MemNet.

IAN [14] contains two LSTMs to encode target words and context information independently and completes the interaction of the two parts of information through the attention mechanism.

TNET [38] proposes a transformation unit for target representation, so that word coding can fully capture the key information of the target. In addition, the authors use a context feature preservation mechanism to better obtain useful information from the context.

TG-SAN [39] includes two core units. One is Structured Context extraction Unit (SCU), which undertakes the task of encoding semantic groups and extracts context fragments related to objects. The second is Context Fusion Unit (CFU), with the purpose of

TABLE 3: Hyperparameters used in the experiment.

Parameter	Value
Dropout rate	0.5
Batch size	32
Learning rate	$2e-5$
Max epoch	20
Max sequence length	128
Optimizer	Adam

learning the contribution of the extracted context to the object.

IMN [22] designs an end-to-end interactive multitask learning network for a variety of fine-grained sentiment analyses. General word vectors and domain-specific word vectors from [40] are concatenated as input. In the model, a special information transfer mechanism is implemented to help the model transfer information between the token level and the document level.

PRET + MULT [21] uses document-level knowledge to improve the performance of aspect-level sentiment analysis. PRET represents the use of documents to train the weight of LSTM, and MULT implies the use of multitask learning methods to complete document-level and aspect-level sentiment analysis tasks.

BERT-FC is the vanilla model built on BERT. The base BERT model is fine-tuned on the target task, and information is extracted at the placeholder [CLS] token for sentiment analysis.

TD-BERT [29] is also based on the BERT fine-tuning model. Instead of the BERT default token [CLS], the vector corresponding to the position of the target term is fed into downstream pipeline. The output is also processed by softmax to get the final emotion category.

AEN-BERT [9] proposes an Attentional Encoder Network (AEN) which does not use the traditional recurrent structure. It employs attention-based encoders for the modeling between a target and its context. They raise the label unreliability issue and introduce label smoothing regularization.

BERT-PT [28] explores a posttraining method on the BERT model with related datasets, with an expectation that the introduction of additional data will improve fine-tuning performance of BERT for sentiment classification.

BERT-pair-QA-M [27] constructs an auxiliary question from the target and uses it together with the original sentence as input. It converts the sentiment classification task into a special QA problem. Since the original paper is applied to task 2 of SemEval-2014, which is not the same as ours, reproduced performance data from [29] is taken as the result.

SK-GCN-BERT [30] proposes a new syntax- and knowledge-based graph convolutional network (SK-GCN) model which leverages the syntactic dependency tree and commonsense knowledge via GCN. In particular, to enhance the representation of the sentence

towards the given aspect, it develops SK-GCN to combine the syntactic dependency tree and commonsense knowledge graph.

SPRN-BERT [31] proposes a semantics perception and refinement network (SPRN) for sentiment analysis with aspects. Local semantic features and global context information are extracted by multichannel convolution and SA, respectively. Then, gated network (DRG) is used to enhance the connection between aspect and context while filtering noise.

5.2. Results and Analysis. Table 4 shows the performance of our model together with previous methods described above. Models in the first part of the table use traditional neural network methods. These methods rely on well-designed attention and LSTM to process static word embeddings. These pretrained word embeddings are generated on large-scale generic corpus or domain-related datasets [40] through the Word2Vec [41] or GloVe [42] method. The second part includes previous multitask learning methods on aspect-level sentiment analysis. The third part shows other models based on the BERT representation. For better performance, they also include customized revisions for the fine-grained sentiment analysis task. Compared with the methods above, our SMLN model shows clear performance improvement over the baseline methods on three datasets from SemEval-2014 and SemEval-2015. This result benefits from the multitask learning setting as well as the information exchange mechanism in CIU.

For the ABSA task, BERT-based models have achieved significant accuracy improvement in comparison to the original static word embeddings. The multilayer Transformer stack structure of BERT has the clear advantage of representing the intrinsic semantics of terms in the context. BERT is a better choice as the shared representation layer in the model, in comparison to static embeddings which lack flexibility in word semantics. When compared to other BERT-based methods, our model still shows significant improvements, about 5% over the baseline BERT-FC model. Our analysis shows that the improvement is mainly from the multitask learning framework. In this work, we introduce opinion terms extraction (OTE) as an auxiliary task. OTE and ABSA tasks are closely related, but there are also clear differences between them. Complementary information from similar but different applications can be used as regularization items between tasks. It effectively improves the generalization performance of the model. In order to further improve the transmission of complementary information between different tasks, we design the CIU module based on an improved self-attention mechanism.

5.3. Ablation Study. In order to study the effects of different components, we gradually add auxiliary tasks and CIU modules starting from the vanilla model. Vanilla model represents the combination of TD-BERT and LSTM network. The experimental results are shown in Table 5, in

TABLE 4: Performance comparison with previous models, with average accuracy as the evaluation metric; the best result for each dataset is boldfaced.

	Method	14Lap	14Res	15Res
Conventional network	TD-LSTM (2016)	68.13	75.63	76.39
	ATAE-LSTM (2016)	68.70	77.20	78.48
	MemNet (2016)	70.33	78.16	77.31
	TNET (2016)	74.65	80.05	78.47
	IAN (2017)	72.10	78.60	78.58
	RAM (2017)	74.49	80.23	79.98
Multitask learning	TG-SAN (2020)	75.27	81.66	—
	PRET + MULT (2018)	71.15	79.11	81.30
BERT	IMN (2019)	75.36	83.89	85.64
	BERT-FC (2018)	76.54	81.28	81.52
	BERT-pair-QA-M (2019)	77.93	85.12	81.89
	AEN-BERT (2019)	78.35	81.46	—
	BERT-PT (2019)	78.07	84.95	—
	TD-BERT (2019)	78.87	85.10	—
Our model	SK-GCN-BERT (2020)	79.00	83.48	83.20
	SPRN-BERT (2021)	79.31	85.03	85.30
	SMLN	80.09	85.67	86.31

TABLE 5: Performance of BERT model with different components, with average accuracy reported over 5 runs.

Model variants	14Lap	14Res
Vanilla model	78.92	85.15
+auxiliary task	79.31	85.33
+CIU	80.09	85.67

which each variant adds a new module based on the previous model. With only the auxiliary task to form a multitask framework, the model achieves a small but noticeable improvement for both the 14Lap and 14Res dataset. It can be considered as the generalization performance improvement brought by the multitask method. At this time, the model cannot benefit from the emotional information provided by the auxiliary task. When the CIU module is added, the additional improvement is about twice that of the previous step. With the CIU, the emotional knowledge has been successfully transferred to the ABSA task.

6. Discussion

For the aspect-based sentiment classification task, we design an SMLN based on multitask learning and attention mechanism. This network can better utilize the rich emotional information in the context and related information among similar tasks at the same time. It tries to solve the problems of sentiment classification and opinion word extraction in an end-to-end manner. In this model, text information is first converted into a vector representation by the BERT preprocessing model. This representation is a common feature in the shared layer that applies to all downstream tasks, and output of the shared layer enters two

independent Bi-LSTM networks to learn the unique features of each task. In particular, this article designs an information interaction unit between two independent representations. This module accomplishes the function of information transfer between the two parts based on the attention mechanism. On publicly available sentiment analysis datasets, its performance is compared to many existing ABSA methods, including some recent work that claims to be state-of-the-art. On all three datasets, the SMLN model achieves competitive results in aspect-based sentiment classification. Its classification accuracy reaches 80.09%, 85.67%, and 86.31% on the 14Lap, 14Res, and 15Res datasets, respectively. To verify the value of its two main components, auxiliary tasks and CIU module, an ablation test is carried out; it shows the step-by-step performance improvement when each individual component is added. The results demonstrate the effectiveness of the SMLN, together with detailed analysis for each component.

In the NLP literature, attention mechanism has been widely used because it can better learn long-range sequence knowledge. However, the latest research shows that the pure self-attention network (SAN) without skip connection and multilayer perceptron (MLP) loses certain expression ability. The loss of feature extraction ability is related to the network depth in double exponential order. Specifically, the researchers prove that the network output converges to a rank-1 matrix at the rate of cubic convergence [43]. Thus, we are currently focusing on the following extensions to the proposed method. First of all, we try to design multilayer attention units in CIU module to obtain stronger feature fusion ability, which is helpful to understand and infer the implied semantics in sentences. Further research aims to explore the changes of internal attention matrix in the process of model reverse updating. Second, we plan to introduce more subtasks into our multitask learning framework, such as entity extraction. The addition of related tasks helps to improve the generalization performance of each task. Finally, we are exploring the effectiveness of our method for other NLP tasks, such as relationship extraction. Overfitting is a common issue for NLP tasks, especially when the model complexity exceeds data size. Multitask learning is an effectively way to improve the generalization ability of a complex model, but understanding the internal correlation between these tasks is more important than blindly stacking more tasks.

Abbreviations

SMLN: Shared Multitask Learning Network
 CIU: Cross Interaction Unit
 LSTM: Long Short-Term Memory
 MTL: Multitask learning
 NLP: Natural language processing
 ASC: Aspect-level sentiment classification
 OTE: Opinion terms extraction
 BERT: Bidirectional encoder representations from transformers.

Data Availability

Restrictions apply to the availability of these data. The datasets SemEval-2014 and SemEval-2015 were taken from <http://alt.qcri.org/semEval2014/task4> and <https://alt.qcri.org/semEval2015>, respectively.

Conflicts of Interest

The authors declare no conflicts of interest.

Acknowledgments

The authors would like to thank Chengdu University of Information Technology for providing the GPU computing power. This research was funded by National Key Research and Development Program of China under grant no. 2017YFC0820700 and Key R&D Projects in Sichuan Province under grant no. 2020YFG0168.

References

- [1] M. Hu and B. Liu, "Mining and summarizing customer reviews," in *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 168–177, ACM, Seattle, Washington D. C., USA, August 2004.
- [2] B. Pang and L. Lee, "Opinion mining and sentiment analysis," *Foundations and Trends in Information Retrieval*, vol. 2, no. 1–2, pp. 1–135, 2008.
- [3] B. Liu, "Sentiment analysis and opinion mining," *Synthesis Lectures on Human Language Technologies*, vol. 5, no. 1, pp. 1–167, 2012.
- [4] M. Pontiki, D. Galanis, J. Pavlopoulos, H. Papageorgiou, I. Androutsopoulos, and S. Manandhar, "Semeval-2014 task 4: aspect based sentiment analysis," in *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pp. 27–35, Dublin, Ireland, August 2014.
- [5] L. Jiang, M. Yu, M. Zhou, X. Liu, and T. Zhao, "Target-dependent twitter sentiment classification," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pp. 151–160, Portland, Oregon, USA, June 2011.
- [6] L. Dong, F. Wei, C. Tan, D. Tang, M. Zhou, and K. Xu, "Adaptive recursive neural network for target-dependent twitter sentiment classification," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, vol. 2, pp. 49–54, Baltimore, Maryland, June 2014.
- [7] D.-T. Vo and Y. Zhang, "Target-dependent twitter sentiment classification with rich automatic features," in *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence*, Buenos Aires, Argentina, June 2015.
- [8] D. Tang, B. Qin, X. Feng, and T. Liu, "Effective lstms for target-dependent sentiment classification," 2015, <https://arxiv.org/abs/1512.01100>.
- [9] Y. Song, J. Wang, T. Jiang, Z. Liu, and Y. Rao, "Attentional encoder network for targeted sentiment classification," 2019, <https://arxiv.org/abs/1902.09314>.
- [10] J. Wagner, P. Arora, S. Cortes et al., "Dcu: aspect-based polarity classification for semeval task 4," in *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pp. 223–229, Dublin, Ireland, August 2014.

- [11] S. Kiritchenko, X. Zhu, C. Cherry, and S. Mohammad, “Nrc-Canada-2014: detecting aspects and sentiment in customer reviews,” in *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pp. 437–442, Dublin, Ireland, August 2014.
- [12] Y. Wang, M. Huang, and L. Zhao, “Attention-based lstm for aspect-level sentiment classification,” in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 606–615, <https://aclanthology.org/people/x/xiaoyan-zhu/>, Austin, TX, USA, November 2016.
- [13] C. Li, X. Guo, and Q. Mei, “Deep memory networks for attitude identification,” in *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pp. 671–680, Cambridge, UK, February 2017.
- [14] D. Ma, S. Li, X. Zhang, and H. Wang, “Interactive attention networks for aspect-level sentiment classification,” 2017, <https://arxiv.org/abs/1709.00893>.
- [15] F. Fan, Y. Feng, and D. Zhao, “Multi-grained attention network for aspect-level sentiment classification,” in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 3433–3442, Brussels, Belgium, October–November 2018.
- [16] S. Ruder, “An overview of multi-task learning in deep neural networks,” 2017, <https://arxiv.org/abs/1706.05098>.
- [17] R. Caruana, “Multitask learning,” *Machine Learning*, vol. 28, no. 1, pp. 41–75, 1997.
- [18] R. Collobert and J. Weston, “A unified architecture for natural language processing: deep neural networks with multitask learning,” in *Proceedings of the 25th International Conference on Machine Learning*, pp. 160–167, Helsinki, Finland, July 2008.
- [19] X. Liu, J. Gao, X. He, L. Deng, K. Duh, and Y.-Y. Wang, “Representation learning using multi-task deep neural networks for semantic classification and information retrieval,” in *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Denver, Colorado, May 2015.
- [20] P. Liu, X. Qiu, and X. Huang, “Recurrent neural network for text classification with multi-task learning,” 2016, <https://arxiv.org/abs/1605.05101>.
- [21] R. He, W. S. Lee, H. T. Ng, and D. Dahlmeier, “Exploiting document knowledge for aspect-level sentiment classification,” 2018, <https://arxiv.org/abs/1806.04346>.
- [22] R. He, W. S. Lee, H. T. Ng, and D. Dahlmeier, “An interactive multi-task learning network for end-to-end aspect-based sentiment analysis,” 2019, <https://arxiv.org/abs/1906.06906>.
- [23] J. Yu and J. Jiang, “Learning sentence embeddings with auxiliary tasks for cross-domain sentiment classification,” in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, Austin, TX, USA, November 2016.
- [24] D. Tang, B. Qin, and T. Liu, “Aspect level sentiment classification with deep memory network,” 2016, <https://arxiv.org/abs/1605.08900>.
- [25] P. Chen, Z. Sun, L. Bing, and W. Yang, “Recurrent attention network on memory for aspect sentiment analysis,” in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 452–461, Copenhagen, Denmark, September 2017.
- [26] P. Zhu and T. Qian, “Enhanced aspect level sentiment classification with auxiliary memory,” in *Proceedings of the 27th International Conference on Computational Linguistics*, pp. 1077–1087, Santa Fe, New Mexico, USA, August 2018.
- [27] C. Sun, L. Huang, and X. Qiu, “Utilizing bert for aspect-based sentiment analysis via constructing auxiliary sentence,” 2019, <https://arxiv.org/abs/1904.02232>.
- [28] H. Xu, B. Liu, L. Shu, and P. S. Yu, “Bert post-training for review reading comprehension and aspect-based sentiment analysis,” 2019, <https://arxiv.org/abs/1904.02232>.
- [29] Z. Gao, A. Feng, X. Song, and X. Wu, “Target-dependent sentiment classification with bert,” *IEEE Access*, vol. 7, Article ID 154299, 2019.
- [30] J. Zhou, J. X. Huang, Q. V. Hu, and L. He, “Sk-gcn: modeling syntax and knowledge via graph convolutional network for aspect-level sentiment classification,” *Knowledge-Based Systems*, vol. 205, Article ID 106292, 2020.
- [31] W. Song, Z. Wen, Z. Xiao, and S. C. Park, “Semantics perception and refinement network for aspect-based sentiment analysis,” *Knowledge-Based Systems*, vol. 214, Article ID 106755, 2021.
- [32] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: pre-training of deep bidirectional transformers for language understanding,” 2018, <https://arxiv.org/abs/1810.04805>.
- [33] A. Vaswani, N. Shazeer, N. Parmar et al., “Attention is all you need,” pp. 5998–6008, 2017, <https://arxiv.org/abs/1706.03762>.
- [34] Y. Wu, M. Schuster, Z. Chen et al., “Google’s neural machine translation system: bridging the gap between human and machine translation,” 2016, <https://arxiv.org/abs/1609.08144>.
- [35] W. Wang, S. J. Pan, D. Dahlmeier, and X. Xiao, “Coupled multi-layer attentions for co-extraction of aspect and opinion terms,” in *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, Hilton San Francisco, CA, USA, February 2017.
- [36] M. Pontiki, D. Galanis, H. Papageorgiou, S. Manandhar, and I. Androutsopoulos, “Semeval-2015 task 12: aspect based sentiment analysis,” in *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pp. 486–495, Denver, Colorado, June 2015.
- [37] W. Wang, S. J. Pan, D. Dahlmeier, and X. Xiao, “Recursive neural conditional random fields for aspect-based sentiment analysis,” 2016, <https://arxiv.org/abs/1603.06679>.
- [38] X. Li, L. Bing, W. Lam, and B. Shi, “Transformation networks for target-oriented sentiment classification,” 2018, <https://arxiv.org/abs/1805.01086>.
- [39] J. Zhang, C. Chen, P. Liu, C. He, and C. W.-K. Leung, “Target-guided structured attention network for target-dependent sentiment analysis,” *Transactions of the Association for Computational Linguistics*, vol. 8, pp. 172–182, 2020, <https://aclanthology.org/2020.tacl-1.12>.
- [40] H. Xu, B. Liu, L. Shu, and P. S. Yu, “Double embeddings and cnn-based sequence labeling for aspect extraction,” 2018, <https://arxiv.org/abs/1805.04601>.
- [41] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representations in vector space,” 2013, <https://arxiv.org/abs/1301.3781>.
- [42] J. Pennington, R. Socher, and C. Manning, “Glove: global vectors for word representation,” in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543, Doha, Qatar, October 2014.
- [43] Y. Dong, J. B. Cordonnier, and A. Loukas, “Attention is not all you need: pure attention loses rank doubly exponentially with depth,” 2021, <https://arxiv.org/abs/2103.03404>.

Research Article

Sentence Classification Using N-Grams in Urdu Language Text

Malik Daler Ali Awan ¹, **Sikandar Ali** ², **Ali Samad** ¹, **Nadeem Iqbal** ³,
Malik Muhammad Saad Missen ¹ and **Niamat Ullah** ⁴

¹Department of Information Technology, Faculty of Computing, The Islamia University of Bahawalpur, 63100 Bahawalpur, Pakistan

²Department of Information Technology, The University of Haripur, 22621 Haripur, Khyber Pakhtunkhwa, Pakistan

³Muhammad Nawaz Shareef University of Agriculture, Multan 61000, Pakistan

⁴Department of Computer Science, University of Buner, 19290 Sawarai Buner, Khyber Pakhtunkhwa, Pakistan

Correspondence should be addressed to Sikandar Ali; sikandar@cup.edu.cn

Received 17 April 2021; Revised 27 May 2021; Accepted 7 November 2021; Published 22 November 2021

Academic Editor: Wei-Chuen Yau

Copyright © 2021 Malik Daler Ali Awan et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

The usage of local languages is being common in social media and news channels. The people share the worthy insights about various topics related to their lives in different languages. A bulk of text in various local languages exists on the Internet that contains invaluable information. The analysis of such type of stuff (local language's text) will certainly help improve a number of Natural Language Processing (NLP) tasks. The information extracted from local languages can be used to develop various applications to add new milestone in the field of NLP. In this paper, we presented an applied research task, "multiclass sentence classification for Urdu language text at sentence level existing on the social networks, i.e., Twitter, Facebook, and news channels by using N-grams features." Our dataset consists of more than 1,00000 instances of twelve (12) different types of topics. A famous machine learning classifier Random Forest is used to classify the sentences. It showed 80.15%, 76.88%, and 64.41% accuracy for unigram, bigram, and trigram features, respectively.

1. Introduction

The text is still dominant and prominent way of communication instead of only pictures, emoji, sounds, and animations. The innovative environment of communication, the real-time availability of the Internet, and the unrestricted communication mode of social networks attracted billions of people around the world. People share insights about various topics, opinions, views, ideas, and events happening around them on social networks in different languages. Social media and news channels: such communication platforms created space for local languages to share information. Google input tool (<https://www.google.com/inputtools/>) provides the language transliteration support to 88 different languages. The development of many local languages supporting tools is another factor that boosted the usage of local languages on social media and news channels. Obviously, people prefer to communicate in local languages instead of global languages

because of easiness in conveying messages. It is also causing to generate heterogeneous data on Internet.

Sifting worthy insights from an immense amount of heterogeneous text of multiple local languages existing on social media is one of the interesting and challenging tasks of Natural Language Processing (NLP). Local language processing certainly provides the invaluable insights to develop NLP applications. These applications can respond in emergencies, outbreaks, and natural disasters, i.e., rain, flood, and earthquake [1]. The interesting feature like real-time interaction of social media has facilitated millions of people to share their intent, appreciation, or criticism [2], i.e., enjoying discount offer by selling brands or criticizing the quality of the product. Extracting and classifying such information are valuable to improve the quality of the product. The implementation of smart cities possesses a lot of challenges, such as decision making, event management, communication, and information retrieval. Extracting useful

insights from an immense amount of text dramatically enhances the worth and quality of smart cities [3]. Similarly, the classified information can be used to predict the effects of the event on the community and take security and rescue measures. Sentence classification information can be used to collect relevant information about the specific topic, top-trends, stories, text summarization, and question and answering system [4, 5]. Such information can be also used to predict upcoming events, situations, and happening. For example, sudden occurrence of earthquake can cause casualties, but classifying such news surely helps us response quickly and save the lives in disasters.

There are many local languages that are being used for communication, i.e., Arabic, Hindi, Persian, Turkish, Urdu, etc. on social media, i.e., Twitter and Facebook. In [6], more than 300 million Urdu language users were reported all around the world. In Pakistan and India, more than 65 million people can speak, understand, and write the Urdu language [7, 8]. The Urdu language is also a national language for different sates in India. It is the national language [9] of Pakistan, which is the 6th populous country in the world (<https://www.worldometers.info/world-population/population-by-country/>). Urdu is widely adopted as a second language all over Pakistan [9, 10]. In other South-Asian countries [11], i.e., Bangladesh, Iran, and Afghanistan also have a considerable number of Urdu language users.

Sentence classification in Urdu language text is a very interesting and challenging task. The lack of resources to classify sentences into different categories is the major challenge. The prominent characteristics of the Urdu language that made the event classification tasks complex and challenging are listed here:

- (i) Cursive language
- (ii) Morphologically rich
- (iii) Different grammatical structure
- (iv) Right to the left scriptwriting style
- (v) No capitalization
- (vi) Lack of resources

The lack of resources, i.e., part of speech tagger (PoS), words stemmer, datasets, and the word annotators are other factors that made Urdu text processing very complex. A considerable amount of Urdu text exists on social networks [12]. There exist only a few referential works on Urdu text. The factors, i.e., huge amount of data, resource poor, and very short referential work, motivated us to explore the Urdu language text. In this research article, we decided to classify sentences into different categories. The purpose of research work is to design a system to extract useful information from Urdu language text and develop various NLP applications.

1.1. Our Contribution

- (i) In this research article, we tried to classify Urdu language text at sentence level in 12 different categories

- (ii) N-grams features, i.e., unigram, bigram, and trigram, are selected to classify sentences
- (iii) We developed multiclass annotated/labeled dataset
- (iv) A dataset larger than others in size (instances) as reported in a state-of-the-art is used to classify sentences

1.2. Our Limitation

- (i) The work is domain-specific (only for Urdu language), but other resource poor languages can be explored in the future
- (ii) It can only classify reported types of events at sentence level

2. Challenges in Sentence Classification for Urdu Language Text

The resource poor languages possess a lot of challenges in the context of resource lacking, i.e., part of speech tagger, annotated datasets, sentence parsers, stemmer, and lexicons. The information extraction related to different events, business, and disasters varies from domain to domain. In the literature, for example, the event was defined in various aspects, such as a verb-, adjective-, and noun-based environmental situation [13, 14]. Extraction and classification of such information require grammatical-, semantic-, and contextual-based information. There are a number of tools for English-like languages that support tackling such challenging task, but the Urdu language is lacking such resources.

Multiclass classification is a type of classification that is the task of automatically assigning the most relevant one class from the given multiple classes (see Figure 1). It also has some serious challenges like detection of sentences that are overlapping in multiple classes [15, 16].

2.1. Limitation of Existing Text Processing Tools for Urdu Language Text. Google language translator (<https://translate.google.com/?hl=en>) supports more than 100 languages. The Urdu language comprises unique structure, complex writing script, and rich morphological features that isolate it from other languages. Urdu is a cursive language written in right to left order and considered as one of the resource-poor languages [8]. It is a mix-composition of different other languages, i.e., Arabic, Persian, Turkish, and Hindi [10]. In contrast to cursive languages, there exists some noteworthy work of information extraction and classification for, i.e., English, French, German, and many other noncursive languages [9, 11].

In the past, researchers were impassive in cursive languages because of poor resources. Therefore, a very low amount of research work exists in cursive language, i.e., Arabic, Persian Hindi, and Urdu [17]. Lack of resources in cursive language was the main barrier to make the research unexcited and vapid [8]. But now, the last few decades' cursive languages have attracted researchers. The main

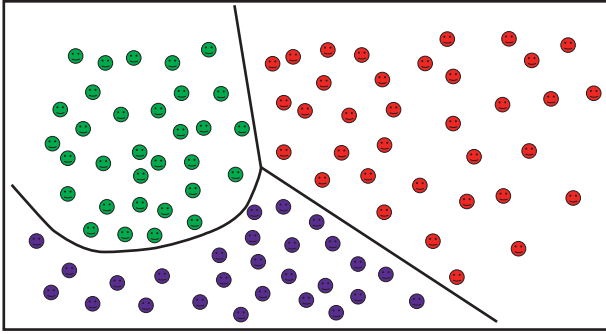


FIGURE 1: Multiple category classification

reason behind the attraction is that a large amount of cursive language data is being generated rapidly on regular basis. Now, some processing tools also have been developed, i.e., part of speech tagger, word stemmer, and annotator that play an important role by making research handier. But these tools are still limited, commercial, and close domain (<http://www.cle.org.pk/>).

Due to the reasons for poor resources, to achieve our goal, we decided to translate Urdu language text into English language text by using Google translator. The output of Google translator is given in Table 1 which depicted that translation is confined to literal translation. It completely misinterpreted the contextual and semantic insights. In example 1, the word (کانٹے کا مقابلہ) kaantay ka muqabla, contest) is translated as (quota), which gives the meaning of (share, حصہ), while the correct word is “contest.” All the bold and italicized words from examples 1 to 6 showed the wrong translation and the limited capability of Google translator. Although, in the past decade, a number of researchers used the strategy of language translation using Google language translation tools and translated the resource poor languages to English language to cope with resources lacking problems, the case of Urdu language is different from other languages. It can be observed in Table 1 that Urdu language cannot be processed by the existing tools.

There is a need to explore and develop resources for the Urdu language to improve the NLP. Table 1 shows that all problems in the Urdu language cannot be resolved by English language processing tools.

2.2. Lack of Recourse. Natural language processing is tightly coupled with resources, i.e., processing resources, datasets, and semantic, syntactical, and contextual information. Textual features; i.e., Part of Speech (PoS) and semantics are important for text processing. Central Language of Engineering (CLE) provides limited access to PoS tagger because of the close domain and paid that diverged the researcher to explore Urdu text.

Contextual features, i.e., grammatical insight (tense) and sequence of words, play important role in text processing. Because of the morphological richness nature of Urdu, a word can be used for a different purpose and convey different meanings depending on the context of contents.

Unfortunately, the Urdu language is still lacking such tools that are openly available for research. Other processing resources, i.e., stemmer, lemmatize, and annotators, are also close domain. Dataset is the core element of research. There is no specific dataset for multiclass sentence classification for Urdu language text. Some datasets for the Urdu language generally exist for name entity extraction with a small number of instances, which are given in Table 2 below.

3. Literature Review

Classification of events from the textual dataset is a very challenging and interesting task of Natural Language Processing (NLP). Textual contents on social media are explored in different ways to extract event information. Generally, the event has been defined as a verb, noun, and adjective [14]. A survey that discussed the various tasks and techniques related to the Urdu language text is available for the researchers as a benchmark text [27]. The event detection is a generic term that is further divided into event extraction and event classification. The lack of resources made research impassive in past to explore the cursive languages like Hindi, Arabic, Persian, and Urdu [28]. A system is designed [29] for Arabic text classification using multiple reduct algorithm. The proposed system showed 94% and 86% accuracies for K-NN and J48 classifiers.

Urdu textual contents explored [30] for classification using the majority voting algorithm. They categorized Urdu text into seven classes, i.e., Health, Business, Entertainment, Science, Culture, Sports, and Wired. They used 21769 news documents for classification and reported 94% precision and recall. The dataset was evaluated using these algorithms, Linear SGD, Bernoulli Naïve Bayes, Linear SVM, Naïve Bayes, random forest classifier, and Multinomial Naïve Bayes. They did not report the overall accuracy of the system for multiple classes. The information about feature selection is also omitted by the researchers, but comparatively, we disclosed the feature selection, engineering, and accuracy of classifiers for multiclass. Our dataset consists of 102960 instances of sentences and twelve (12) classes that are comparatively very greater.

A framework [31] proposed a tweet classification system to rescue people looking for help in a disaster like a flood. The developed system was based on the Markov Model achieving 81% and 87% accuracy for classification and location detection, respectively. The features used in their system are as follows:

- (i) Number of words in a tweet (w)
- (ii) Verb in a tweet by (verb)
- (iii) Number of verbs in a tweet by (v)
- (iv) Position of the query by (Pos)
- (v) Word before query word (before)
- (vi) Word after query word (after)

A neural network-based system that is a combination of conventional neural network and recurrent neural network was designed to extract events from English, Tamil, and

TABLE 1: Wrong translation example of Google Translator.

Sr. No.	Urdu	English
1	بھارت اور انگلینڈ کے درمیان کانٹے کا مقابلہ جاری ہے	The quota is going on between England and India.
2	بجٹ بل کو لے کے حکومت اور اپوزیشن میں ٹھن گئی	Government and opposition were <i>tied</i> to take the budget bill.
3	عیدالضحیٰ پر مسلمان جانوروں کی قربانی کرتے ہیں	Muslims <i>sacrifice sacrifices</i> on the occasion.
4	ایران پر امریکی پابندیوں کے بعد بھارت نے بھی ایران سے تیل کی درآمد بند کر دی	<i>After Iran's sanctions on Iran</i> , India also stopped importing oil from Iran.
5	شہید لیفٹیننٹ کرنل راشد کریم بیگ کی نماز جنازہ گلگت میں ادا کر دی گئی	The funeral prayers of martyr Lieutenants Colonel Rashid Karim Bag were <i>released</i> in Gilgit.
6	دھرتی ماں پر جان قربان کرنے والے جان باز قوم کے ہیرو ہیں	<i>On the forehead mother is the hero of the people who sacrificed life.</i>

TABLE 2: Urdu language dataset.

Sr. No.	Dataset
1	Enabling Minority Language Engineering (EMILLE) (only 200000 tokens) [18]
2	Becker-Riaz corpus (only 50000 tokens) [19]
3	Computing Research Laboratory (CRL) annotated corpus (only 55,000 tokens are publicly available data corpora) [20]
4	International Joint Conference on Natural Language Processing (IJCNLP) workshop corpus (only 58252 tokens)
5	Urdu Named Entity Recognition (UNER) [4]
6	Corpus of 705 sentences [21]
7	Corpus of BBC Urdu, Daily Jang [22]
8	corpus of 19.3 million words [23]
9	COUNTER, Naïve, NPUU [24, 25]
10	DSL Urdu news [26]

Hindi languages. It showed f-score 39.91%, 37.42%, and 39.71% [32].

To classify Urdu news headlines [23] by using maximum indexes of vectors, the stemmed and nonstemmed textual data was used for experiments. The system was specifically designed for text classification instead of sentence classification. Their proposed system achieved 78.0% for competitors and 86.6% accuracy for the proposed methodology. In comparison, we used sentences of Urdu language for classification and explored the textual features of sentences. A multiclass event classification task [18] was performed for Urdu language text that evaluated the performance of different classifiers. On the contrary, we evaluated the performance of Random Forest classifiers for different level of n-gram features.

Twitter [19] was used to detect natural disasters, i.e., bush fires, earthquakes, and cyclones, as well as humanitarian crises. To be aware of emergencies situation in natural disasters a framework work designed based on SVM and Naïve Bayes classifiers using word unigram, bigram, length, number of #Hash tag, and reply. These features were selected on sentence bases. SVM and Nave Bayes showed 87.5% and 86.2% accuracy, respectively, for tweet classification, i.e., seeking help, offering for help, and none. An intent mining system was developed [2] to facilitate citizens and cooperative authorities using a bag of the token model. The researchers exploited the hybrid feature representation for binary classification and multilabel classification. It showed a 6% to 7% improvement in the top-down feature set processing approach. Intelligence information retrieval plays a vital role in the management of smart cities [33]. This

information helps enhance security and emergency management capabilities in smart cities. A very popular social website's twitter textual data used [20] to extract and classify events for the Arabic language. Implementation and testing of Support Vector Machine (SVM) and Polynomial Network (PN) algorithms showed promising results for tweet classification 89.2% and 92.7%. Stemmer with PN and SVM magnified the classification by 93.9% and 91.7%, respectively. Social events [30] were extracted assuming that, for prediction, either parties or one of them is aware of the event. The research aimed to find the relation between related events. Support Vector Machine (SVM) with kernel method was used on adopted annotated data of Automated Content Extraction (ACE). Structural information derived from the dependency tree and parsing tree is utilized to derive new structures that played important role in event identification and classification. In [24], Urdu text classification deep learning models evaluated using existing benchmark datasets [25]. The classification is performed for small, medium, and large size of preexisting dataset for product analysis [24]. A benchmark for the Urdu text classification [26] presented the comparison of machine learning classifiers using n-gram features on two closed source benchmark datasets CLE Urdu Digest 1000k, and CLE Urdu Digest 1Million and publicly available dataset.

A research study [34] was conducted to evaluate the students teaching environment using deep learning classifier RNN. The dataset consists of 15 4000 instructor reviews. Deep learning and conventional machine learning classifiers are evaluated on the dataset. Deep learning classifiers using word embedding ensemble with attention mechanism and

the system showed 98.29% accuracy. In the research article [21], linguistic and psychological features sets are used to analyze the sentiment on twitter. Five linguistic categories and their ensembles were used as input and four supervised classifiers evaluated. The analysis showed that ensemble models are better in performance than conventional classifiers [21]. The authors reported [34] that software products and organization performance were analyzed using k-mean and parallel k-mean clustering to improve the educational environment. The evaluation experiment was performed on 10,000 to 5,000 numerical instances. The results analysis showed that parallel clustering improved the time elapsed. A sentiment analysis performed to analyze the usage of product. The reviews given on product were used for sentiment classification. The weighted word embedding and deep neural networks are used in combination. The combined architecture of TF-IDF and CNN-LSTM showed better results as compared to conventional deep learning models [40]. The AdaBoost and Naïve Bayes showed the highest accuracy 88.1% with combination of consistency features. To classify unstructured data, hybrid supervised clustering based on ensemble scheme is used to compare the conventional and ensemble classifiers [34]. A feature selection model was introduced to classify text that is based on generic ranking aggregation [34].

4. Sentence Classification Methodology

Textual data classification possesses a lot of challenges, i.e., word similarity, poor grammatical structure, miss-use of terms, and multilingual words. We decided to adopt a supervised classification approach to classify Urdu sentences into different categories.

Sentence classification for Urdu Language text is performed by supervised machine learning approach. A complete overview of the multiclass sentence classification methodology is given here in Figure 2. The proposed framework is depicted in Figure 3.

4.1. Data Collection. Urdu textual data is collected from popular social networks, i.e., twitter, famous news channel's blogs, i.e., Geo News, Urdu Point, and BBC Urdu. Data collection consists of a title, body, published date, location, and URL. In the phase of data collection, a PHP-based web scraper is used to crawl data from the above-cited social websites. A complete post is retrieved from the websites and stored in MariaDB (database). As we have described earlier, our task is to classify events at sentence level instead of whole document classification. Our dataset consists of more than one million (102, 960) label sentences of different types of events. All the different types of events used in our research work and their maximum number of instances are shown in Figure 4.

There are twelve different types of events that we try to classify in our research work. These events are a factual representation of the state of the people. In Figure 2, the imbalances number of instances of each event is given. It can be visualized that politics, sports, and fraud and corruption

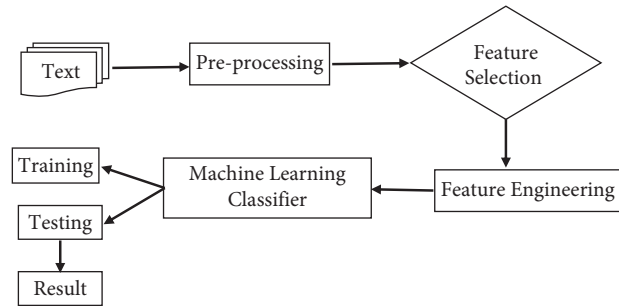


FIGURE 2: Generic abstract diagram of proposed system.

have a higher number of instances, while inflation, sexual assault, and terrorist attack have a lower number of instances. These imbalances number of instances made our classification more interesting and challenging.

In case of multiclass sentence classification, the corpus comprises many classes. There are different types of events used in our research work, i.e., sports, Inflation, Murder & Death, Terrorist Attack, Politics, Law and Order, Earthquake, Showbiz, Fraud and Corruption, Weather, Sexual Assault, and Business. All the sentences of the dataset are labeled by the above mentioned twelve (12) different types of events. Finally, a numeric (integer) value is assigned to each type of event label (see Table 3 for more details on the label and its relevant numeric value).

4.2. Preprocessing. The initial steps are performed on the corpus to prepare machine learning algorithms, because textual data cannot be directly processed by machine learning classifiers. It also contains many irrelevant words. So, we must apply some preprocessing steps; stemming is a powerful technique in preprocessing to find the root words and reduce the feature space. But, in our case, the nature of dataset is entirely different, because our dataset is a mix-up of novel/rare event and common events. Furthermore, generally, each type of event has varying vocabulary of text. Therefore, we assumed/considered that stemming would not affect the performance; that is why there is no need to use stemmer. The details of all the preprocessing steps followed in our research problem to prepare the dataset are given in Figure 5.

4.2.1. Post Splitting. The PHP crawler extracted the body of the post. It comprises many sentences as a paragraph. In the Urdu language script, sentences end with a sign called “-” Hyphen (Khatma-ہم). It is a standard punctuation mark in the Urdu language to represent the end of the sentence. As mentioned earlier, we are performing event classification at the sentence level. So, we split paragraphs of every post into sentences. Every line in the paragraphs ending at Hyphen is split as a single line.

4.2.2. Stop Words Elimination. Generally, those words that occur frequently in text corpus are considered as stop words. These words merely affect the performance of the classifier.

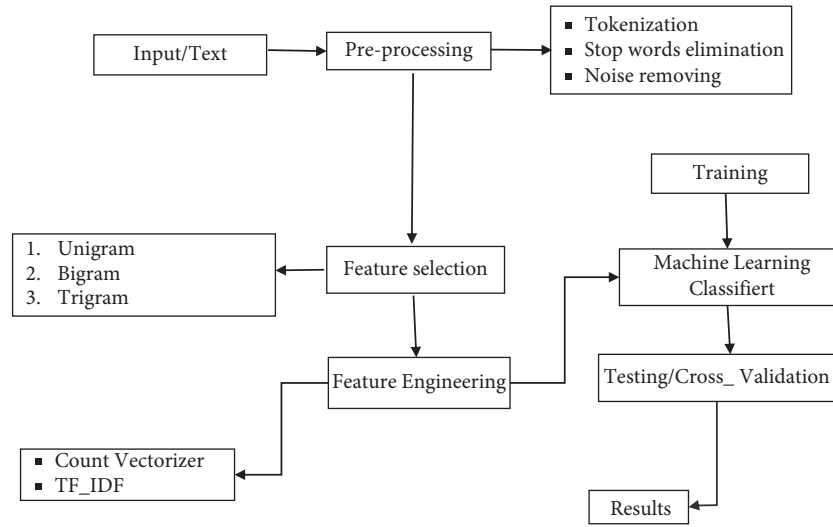


FIGURE 3: Flow diagram of proposed methodology for sentence classification from Urdu language text.

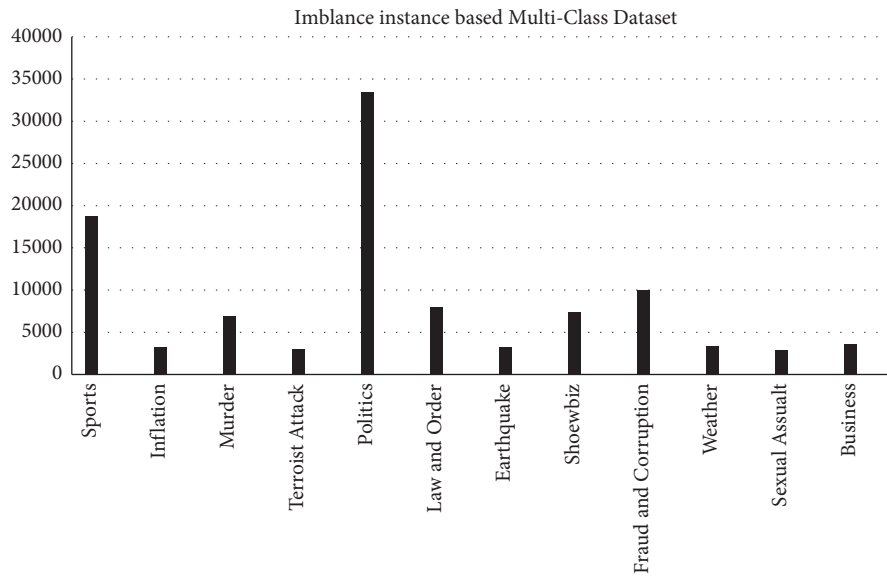


FIGURE 4: Maximum number of instances.

TABLE 3: Sentence with annotated label.

Sentence	Label
Sports	1
Inflation	2
Murder and Death	3
Terrorist Attack	4
Politics	5
Law and Order	6
Earthquake	7
Showbiz	8
Fraud and Corruption	9
Rain/Weather	10
Sexual Assault	11
Business	12

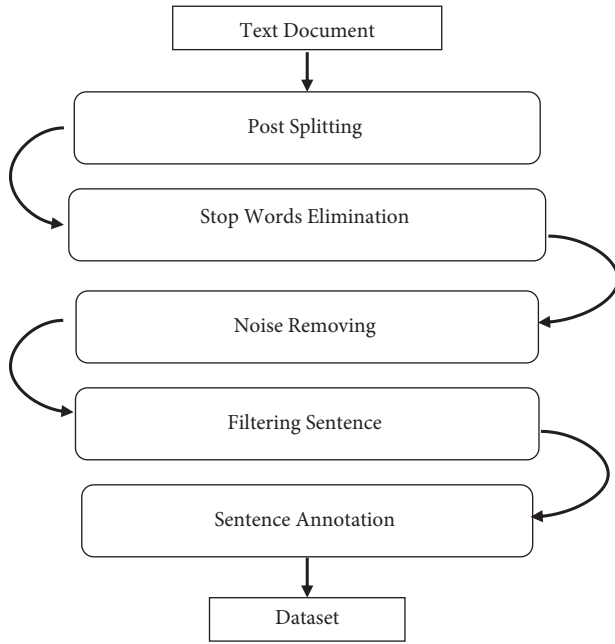


FIGURE 5: Dataset preprocessing steps.

The punctuation marks (“!”, “@”, “#”, etc.) and frequent words of the Urdu languages (کے، کی، وغیرہ) are the common examples of stop words [33] that do not play an influential role in event classification for the Urdu language text are eliminated from the corpus. Stop words elimination reduces the memory and processing utilization and makes the processing efficient.

4.2.3. Noise Removal. Our data is collected by different sources (see Section 4.1). It contains a lot of noisy elements, i.e., multilanguage words, links, mathematical characters, special symbols, etc. In collected corpus, we found many multilingual sentences in the post. To make our corpus clean and ready for further processing, we removed those sentences, irrelevant links, and special characters from the corpus.

4.2.4. Filtering Sentences. The nature of our problem confined us to define the limit of words per sentence. Because of the multiple types of events, it is probably hard to find the sentence of the same length. We decided to keep the maximum number of sentences in our corpus. All those sentences, which are very short and very long, are removed from our corpus. We observed that a lot of sentences vary in length from 5 words to 250 words. We decided to use sentences that consist of 5 words to 150 words to lemmatize our research problem and consumption of processing resources.

4.2.5. Sentence Labeling. In supervised learning, providing output (Label) details in the corpus is a core element. Sentence labeling is an exhausting task that requires deep knowledge and an expert’s skill of language. All the

sentences were manually labeled by observing the title of the post and body of sentences by Urdu language experts (see Table 2). Three Urdu language experts were engaged in the task of sentence labeling. One of them is Ph.D. (Scholar), while the other two are M.Phil. To the best of our knowledge, it is the first largest labeled dataset for the multiclass event in the Urdu language. It consists of more than 1,00,000 label instances.

4.3. Feature Selection. The performance of prediction or classification models is cohesively related to the appropriate feature selection. Features selection in machine learning is very important to develop accurate models. Features are fundamental parameters that are given as input to learning classifiers in the phase of model development. The trained model further can be tested to classify, predict, or assign labels to new instances. A sentence contains very limited information that is insufficient to differentiate among multiple sentences. Classification of text at sentence level requires the contextual information. To capture the contextual information of sentences, we decided to use unigram, bigram, and trigram features for text classification at sentence level. The examples of the proposed features are given in Table 4. For example, the sentence “The brutal attack of Covid-19 killed billion of people” can be converted to n -gram features after preprocessing, as shown below:

4.3.1. Feature Engineering. Feature engineering is a way of generating specific features from a given set of features and converting selected features to machine-understandable format.

Our dataset that is text-based consists of more than 1,00,000 labeled instances, i.e., sports, inflation, death, terrorist attack, sexual assault, etc. For the 12 classes, we generated three features, i.e., unigram, bigram, and trigram. All the textual features are converted to numeric format using (Term Frequency_ Inverse Document Frequency) TF_IDF. The scikit-learn package is used to transform text data into numerical value [17].

4.3.2. Term Frequency Inverse Document Frequency. It is a statistical measure of word w to understand the importance of that word for specific document d in the corpus. The importance of w is proportionally related to frequency; i.e., the higher the frequency, the more important. The mathematical formulas of the TF_IDF are given below:

Term Frequency (TF) counts the number of terms how often it appears in the document. The formula of term frequency is given as follows:

$$TF = \frac{\text{Number of time term } t \text{ appears in a document}}{\text{Total number of term } t \text{ in the document}} \quad (1)$$

The inverse document frequency is used to identify that a term is rare or common in the corpus. The formula is given here:

TABLE 4: Selected features.

Sr. No.	Feature_Name	Example
1	Unigram	“Brutal,” “attack”, “Covid-19,” “killed,” “billion,” “people”
2	Bigram	“Brutal attack,” “attack Covid-19,” “Covid-19 killed,” “killed billion,” “billion people”
3	Trigram	“Brutal attack Covid-19,” “attack Covid-19 killed,” “Covid-19 killed billion,” “killed billion people”

$$IDF_t = \text{Log}_e \frac{\text{Total number of the documents}}{\text{Total number of the documents term } t \text{ appears}} \quad (2)$$

The TF-IDF consists of the product of two components, i.e., term frequency and inverse document frequency.

$$TF_IDF = TF * IDF. \quad (3)$$

4.4. Training Dataset. To develop a generic model for event classification, we divided our dataset into three subsets, i.e., training dataset, testing, and validation dataset.

Random distribution of data is performed by using python library scikit. We distributed a 70% dataset randomly for training purposes. There are 72072 labeled instances for different types of events in our training dataset. A multiclass-instances-based training dataset is used for training deep learning models to develop a generic model.

4.5. Testing Dataset. To evaluate the performance of our proposed framework, we used a 30% dataset for testing/validations purposes. It consists of 30888 unknown instances that are never seen by trained models.

5. Machine Learning Classifier

Classifiers are the algorithms used to classify data instances. In machine learning for textual data, many classifiers exist, but, in our research work, we decided to use the Random Forest for classification, because it consists of multiple decision trees that are based on rules. Furthermore, it has never been used for text classification at the sentence level for the Urdu language text.

5.1. Random Forest. A multiclass classifier that is based on a large amount of imbalance dataset. It is a meta learner having multiple trees that form the forest.

Overall classification in random forest is determined by the vote of random trees. Vote of trees is used to assign the specific class to the input (see Figure 6). It follows a bootstrapping-like technique in the training phase. One-third of instances are preserved in out-of-bag. Features are chosen randomly for each tree. Finally, out-of-bag instances are used to test the model. Average misclassification of overall trees is known as estimated errors that can be used to measure the performance of classifier [6].

5.2. Performance Measuring Parameters. The most common performance measuring [29] parameters, i.e., precision, recall, and f1_measure, are used to evaluate the proposed framework.

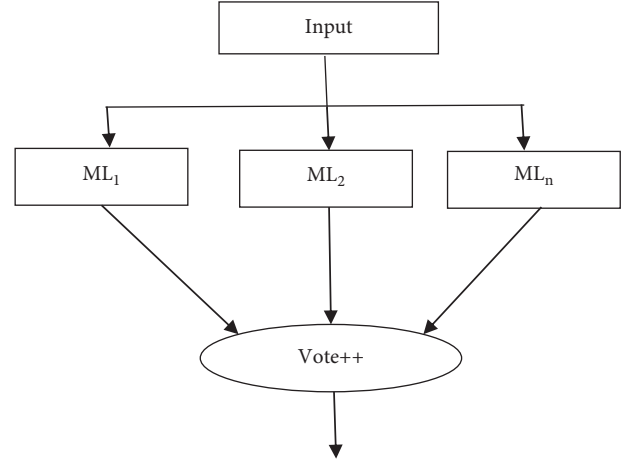


FIGURE 6: Random forest

5.2.1. Precision. Precision: it is the measurement of the exactness of the classifier. The precision calculating formula is given as follows:

$$\text{Precision} = \frac{TP}{TP + FP}. \quad (4)$$

5.2.2. Recall. Recall: it measures the completeness of the classifier results. It is calculated by the following equation:

$$\text{Precision} = \frac{TP}{TP + FN}. \quad (5)$$

5.2.3. F1_Measure. F1_Measure is the harmonic mean of precision and recall and can be calculated as

$$F1 = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \quad (6)$$

5.2.4. Accuracy. Accuracy: it is the most common measure for classifier performance and can be calculated as follows:

$$\text{Accuracy} = \frac{(TP + TN)}{(TP + TN + FP + FN)}. \quad (7)$$

6. Experimental Results

To evaluate our dataset, the Python package scikit-learn is used to perform text classification at the sentence level. Comparison of the results obtained by using the proposed n-grams features is given below.

TABLE 5: Unigram

Label	Event	Precision	Recall	F1_Measure
1	Sports	0.94	0.93	0.93
2	Inflation	0.93	0.97	0.95
3	Murder and Death	0.71	0.62	0.67
4	Terrorist Attack	0.78	0.55	0.65
5	Politics	0.71	0.90	0.79
6	law and order	0.66	0.40	0.50
7	Earthquake	1.00	1.00	1.00
8	Showbiz	0.93	0.81	0.86
9	Fraud and corruption	0.75	0.58	0.65
10	Rain/weather	0.94	0.98	0.96
11	Sexual Assault/Intercourse	0.96	0.98	0.97
12	Business	0.84	0.63	0.72
Overall accuracy (%)		80.15		

TABLE 6: Bigram.

Label	Event	Precision	Recall	F1_Measure
1	Sports	0.92	0.89	0.91
2	Inflation	0.92	0.96	0.94
3	Murder and Death	0.70	0.55	0.62
4	Terrorist Attack	0.51	0.54	0.52
5	Politics	0.70	0.86	0.77
6	law and order	0.58	0.39	0.47
7	Earthquake	0.99	1.00	1.00
8	Showbiz	0.88	0.72	0.79
9	Fraud and corruption	0.69	0.55	0.61
10	Rain/weather	0.93	0.97	0.95
11	Sexual Assault/Intercourse	0.93	0.98	0.96
12	Business	0.78	0.64	0.71
Overall accuracy (%)		76.88		

TABLE 7: Trigram.

Label	Event	Precision	Recall	F1_Measure
1	Sports	0.44	0.96	0.60
2	Inflation	0.92	0.96	0.94
3	Murder and Death	0.68	0.39	0.49
4	Terrorist Attack	0.67	0.38	0.49
5	Politics	0.76	0.61	0.68
6	law and order	0.56	0.30	0.39
7	Earthquake	1.00	1.00	1.00
8	Showbiz	0.91	0.45	0.60
9	Fraud and corruption	0.70	0.46	0.56
10	Rain/weather	0.94	0.96	0.95
11	Sexual Assault/Intercourse	0.95	0.98	0.96
12	Business	0.76	0.47	0.58
Overall accuracy (%)		64.41		

In Table 5, we present the performance measuring parameters of different types of sentences. The Random Forest (RF) classifier showed 80.15% accuracy using unigram feature.

We also evaluated the performance of Random Forest classifier for bigram features to enhance the accuracy of the system. But bigram showed lower results as compared to unigram. The overall accuracy using bigram is 76.88% presented in Table 6.

We further explored the trigram features, but the accuracy of classifiers was decreasing. The trigrams features showed very low results as compared to unigram and bigram features (see Table 7 for more details). The machine learning classifier Random Forest showed 64.41% overall accuracy.

The comparison of accuracy of all features, i.e., unigram, bigram, and trigram, is given in Figure 7.

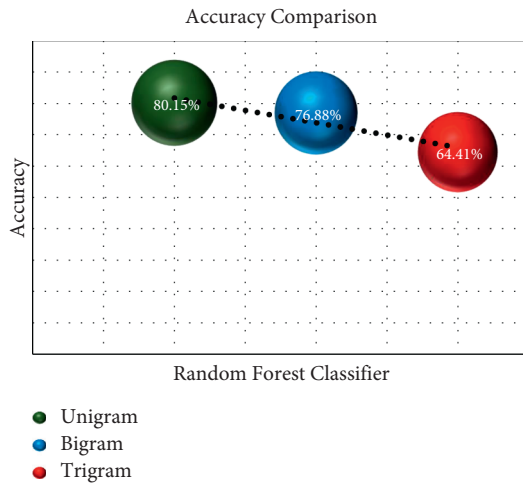


FIGURE 7: Radom forest accuracies using unigram, bigram, and trigram features

7. Conclusion and Future Work

In a comprehensive review of Urdu literature, we found a few numbers of referential works related to Urdu text processing. The main hurdle in Urdu exploration is the unavailability of the processing resources i.e., eventual dataset, close-domain Part of speech tagger, lexicons, and other supporting tools.

A massive amount of Urdu textual data exists on social networks and news websites. Multiclass classification for the Urdu language text at the sentence level is performed by selecting appropriate features. N-grams are the key features to achieve our expected results, because they can retain the sequence and contextual information of sentence. Count_Vectorizer and TF-IDF feature generating methods are used to convert text into numeric real value for machine learning models. We did not use the word2vec model because of lacking pretrained models. Furthermore, customized pretrained models that are prepared using the corpus in hand are very inefficient in context of accuracy. The reason is that the amount of data is insufficient to build such model (Custom Word2Vec model).

Urdu event dataset was used to evaluate Random Forest using unigram, bigram, and trigram features. In our proposed framework, Random Forest showed Unigram, bigram, and trigram accuracy of 80.15% 76.88%, and 64.41%, respectively.

- (i) Many open-source tools, i.e., PoS tagger, annotation tools, event datasets, and lexicons, can be created to extend the research areas in the Urdu language.
- (ii) In the future, many other types of events of other domains like the medical event, social, local, and religious events can be classified using an advanced form of machine learning, i.e., deep learning.
- (iii) In the future, grammatical, contextual, and lexical information can be used to categorize events. Temporal information related to sentence can be further utilized to classify it as real and retrospective.

- (iv) Classification of Urdu language text can be performed at the document level and phrase level.
- (v) Deep learning classifiers can be used for a many other types of sentences.

Data Availability

The data collected during the data collection phase are available from the corresponding authors upon request.

Conflicts of Interest

The authors declare that they have no potential conflicts of interest.

Acknowledgments

The authors thank Dr. Mujtaba Hasnain, Department of Information Technology, Faculty of Computing, The Islamia University of Bahawalpur, 63100 Bahawalpur, Pakistan.


References

- [1] J. Yin, S. Karimi, A. Lampert, M. Cameron, B. Robinson, and R. Power, "Using social media to enhance emergency situation awareness," in *Proceedings of the Twenty-fourth international joint conference on artificial intelligence*, Buenos Aires, Argentina, 2015 June.
- [2] H. Purohit, G. Dong, V. Shalin, K. Thirunarayan, and A. Sheth, "Intent classification of short-text on social media," in *Proceedings of the 2015 IEEE International Conference on Smart City/SocialCom/SustainCom (SmartCity)*, pp. 222–228, IEEE, Chengdu, China, 2015, December.
- [3] M. Alkhatibl, M. El Barachi, and K. Shaalan, "Using Arabic social media feeds for incident and emergency management in smart cities," in *Proceedings of the 2018 3rd International Conference on Smart and Sustainable Technologies (SpliTech)*, pp. 1–6, IEEE, Split, Croatia, 2018 June.
- [4] W. Khana, A. Daudb, J. A. Nasira, and T. Amjada, "Named entity dataset for Urdu named entity recognition task," *Organization*, vol. 48, p. 282, 2016.
- [5] I. S. Jacobs and C. P. Bean, "Fine particles, thin films and exchange anisotropy (effects of finite dimensions and interfaces on the basic properties of ferromagnets)," in *Spin Arrangements and Crystal Structure, Domains, and Micromagnetics*, G. T. Rado and H. Suhl, Eds., vol. 3, pp. 271–350, Academic, New York, NY, USA, 1963.
- [6] F. Livingston, "Implementation of Breiman's random forest machine learning algorithm," *ECE591Q Machine Learning Journal Paper*, pp. 1–13, 2005.
- [7] M. Naz and S. Hussain, "Binarization and its evaluation for Urdu Nastalique document images," in *Proceedings of the Multi Topic Conference (INMIC), 2013 16th International*, pp. 213–218, IEEE, Lahore, Pakistan, 2013 December.
- [8] S. Mukund, R. Srihari, and E. Peterson, "An information-extraction system for Urdu---a resource-poor language," *ACM Transactions on Asian Language Information Processing*, vol. 9, no. 4, p. 15, 2010.
- [9] D. Nadeau and S. Sekine, "A survey of named entity recognition and classification," *Linguisticæ Investigationes. International Journal of Linguistics and Language Resources*, vol. 30, no. 1, pp. 3–26, 2007.

- [10] S. M. Ghulam and T. Rahim Soomro, "Twitter and Urdu," in *Proceedings of the 2018 International Conference on Computing, Mathematics and Engineering Technologies (iCoMET)*, pp. 1–6, IEEE, Sukkur, Pakistan, 2018.
- [11] K. Riaz, "Concept search in Urdu," in *Proceedings of the 2nd PhD workshop on Information and Knowledge Management*, pp. 33–40, Napa Valley, CA, USA, 2008, October.
- [12] S. T. Ghulam and T. Rahim Soomro, "Twitter and Urdu," Department of Computer Science SZABIST Dubai Campus Dubai United Arab Emirates (UAE).
- [13] S. Ramesh and S. Kumar, "Event extraction from natural language text," *International Journal of Engineering Sciences & Research Technology*, 2016.
- [14] J. P. Singh, Y. K. Dwivedi, N. P. Rana, A. Kumar, and K. K. Kapoor, "Event classification and location prediction from tweets during disasters," *Annals of Operations Research*, vol. 283, pp. 1–21, 2017.
- [15] X. Kong, X. Shi, and P. S. Yu, "Multi-label collective classification," in *Proceedings of the 2011 SIAM International Conference on Data Mining*, pp. 618–629, Society for Industrial and Applied Mathematics, Mesa, AZ, USA, 2011, April.
- [16] A. Sarker and G. Gonzalez, "Portable automatic text classification for adverse drug reaction detection via multi-corpus training," *Journal of Biomedical Informatics*, vol. 53, pp. 196–207, 2015.
- [17] F. Pedregosa, G. Varoquaux, A. Gramfort et al., "Scikit-learn: machine learning in Python," *Journal of Machine Learning Research*, vol. 12, no. Oct, pp. 2825–2830, 2011.
- [18] D. Ali, M. M. S. Missen, and M. Husnain, "Multiclass event classification from text," *Scientific Programming*, vol. 2021, Article ID 6660651, 15 pages, 2021.
- [19] A. Agarwal and O. Rambow, "Automatic detection and classification of social events," in *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pp. 1024–1034, Association for Computational Linguistics, Cambridge, MA, USA, 2010 October.
- [20] S. Hussain, *Resources for Urdu Language Processin*, Center for Research in Urdu Language Processing National University of Computer and Emerging Sciences B Block, Faisal Town, Pakistan.
- [21] A. Onan, "Sentiment analysis on Twitter based on ensemble of psychological and linguistic feature sets," *Balkan Journal of Electrical and Computer Engineering*, vol. 6, no. 2, pp. 69–77, 2018.
- [22] T. Sakaki, M. Okazaki, and Y. Matsuo, "Earthquake shakes Twitter users: real-time event detection by social sensors," in *Proceedings of the 19th international conference on World wide web*, pp. 851–860, ACM, Raleigh North, CA, USA, 2010 April.
- [23] A. R. Ali and M. Ijaz, "Urdu text classification," in *Proceedings of the 7th international conference on frontiers of information technology*, p. 21, 2009 December.
- [24] M. P. Akhter, Z. Jiangbin, I. R. Naqvi, M. Abdelmajeed, and M. Fayyaz, "Exploring deep learning approaches for Urdu text classification in product manufacturing," *Enterprise Information Systems*, pp. 1–26, 2020.
- [25] M. P. Akhter, Z. Jiangbin, I. R. Naqvi, M. Abdelmajeed, A. Mehmood, and M. T. Sadiq, "Document-level text classification using single-layer multisize filters convolutional neural network," *IEEE Access*, vol. 8, pp. 42689–42707, 2020.
- [26] M. N. Asim, M. U. Ghani, M. A. Ibrahim, W. Mahmood, A. Dengel, and S. Ahmed, "Benchmarking performance of machine and deep learning-based methodologies for Urdu text document classification," *Neural Computing & Applications*, pp. 1–33, 2020.
- [27] A. Daud, W. Khan, and D. Che, "Urdu language processing: a survey," *Artificial Intelligence Review*, vol. 47, no. 3, pp. 279–311, 2017.
- [28] N. Alsaedi and P. Burnap, "Arabic Event Detection in Social Media," *Cardiff School of Computer Science and Informatics*, Cardiff University, Cardiff, CF, UK.
- [29] Q. A. Al-Radaideh and M. A. Al-Abrat, "An Arabic text categorization approach using term weighting and multiple reducts," *Soft Comput*, vol. 23, no. 14, pp. 5849–5863, 2018.
- [30] M. Usman, Z. Shafique, S. Ayub, and K. Malik, "Urdu text classification using majority voting," *International Journal of Advanced Computer Science and Applications*, vol. 7, no. 8, pp. 265–273, 2016.
- [31] A. Kuila, S. chandra Bussa, and S. Sarkar, "A neural network based Event extraction system for Indian languages," *Fire*, vol. 2266, 2018.
- [32] K. Ahmed, M. Ali, S. Khalid, and M. Kamran, "Framework for Urdu News Headline Classification," *Journal of Applied Computer Science & Mathematics*, vol. 10, 2016.
- [33] M. Alkhatib, M. El Barachi, and K. Shaalan, *Using Arabic Social Media Feeds for Incident and Emergency Management in Smart Cities*, University of Wollongong in Dubai, Faculty of Engineering and Information Sciences, Knowledge Village, Dubai, UAE.
- [34] R. Shang, B. Ara, I. Zada, S. Nazir, Z. Ullah, and S. U. Khan, "Analysis of simple K-mean and parallel K-mean clustering for software products and organizational performance using education sector dataset," *Scientific Programming*, vol. 2021, Article ID 9988318, 20 pages, 2021.

Research Article

The Model of Makerspace Development Element and Performance Analysis Based on NVivo Classification

Yingyan Wang^{1,2} and Rui Zeng¹ 

¹Yiwu Industrial and Commercial College, Yiwu 322000, China

²Yiwu Innovation Research Institute, Yiwu 322000, China

Correspondence should be addressed to Rui Zeng; zengrui@ywicc.edu.cn

Received 19 July 2021; Revised 2 September 2021; Accepted 13 September 2021; Published 5 November 2021

Academic Editor: KunHong Liu

Copyright © 2021 Yingyan Wang and Rui Zeng. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Makerspace is an innovation and entrepreneurship service platform. Its ultimate goal is to realize value cocreation among multiple subjects. The development of Makerspace and the realization of value are closely related to the Makerspace Development Policy. With the help of NVivo/ software, we analyzed 17 Makerspace Development Policies. Through the selection of samples, statistical analysis of word frequency, and coding analysis, the 7 core elements of entrepreneurial incentives, fee reductions and exemptions, employment support, entrepreneurial environment, talent support, financial support, and service optimization are summarized to form an empirical research survey of the development policy elements of the makerspace project. The author uses the game evolution method to analyze the mechanism of the internal relationship of innovation factors, builds a model of the development factors of the makerspace, selects 52 makerspace companies to conduct a questionnaire survey, using the model to conduct empirical research on the performance of existing policies, and finally proposes a model for makerspace development.

1. Introduction

Policy is the basic tool for governing the country, and public policy is the main means for the state and the government to realize their functions [1]. To comply with the new trend of mass innovation and entrepreneurship and further give play to its supporting role in the country's economic development, the State Council issued the *Guiding Opinions on Developing Makerspace and Promoting Mass Innovation and Entrepreneurship* in March 2015, which officially kicked off the prelude to the construction of makerspace. To further improve the service level of makerspace and promote the development of the new normal of China's economy, the State Council issued the *Opinions of the State Council on Promoting High-Quality Development of Innovation and Entrepreneurship for an Upgrade Version* in September 2018, which officially marked the start of a stage of high-quality construction of makerspace. Driven by China's innovation-driven development strategy decision-making, provincial

governments have issued a series of policies to accelerate the development of makerspace and improve the quality of innovation and entrepreneurship services.

In recent years, the research of makerspace policy has gradually become a hot topic of government decision-making departments and academia. Scholars interpret and analyze the policy of makerspace from different perspectives. Poter from the perspective of industrial clusters combined with diamond theory carried out a simple analysis of the feasibility of the government to achieve agglomeration of elements. The main ways of action are as follows: (1) The government can influence the diamond system through subsidies, education, and financial market policies. The way in which the factors of production in the market then affect market demand affects the concentration of factors. (2) The government can affect the structural strategy of the enterprise and the form of competitors by affecting the industrial development environment. (3) The industrial form, external market conditions, and production factors in turn also

affects policies [2]. Jacobs analyzed government behavior in industrial agglomeration under urban diversification and believed that the government can play a role through the allocation of capital and urban planning [3, 4]. Baldwin and Fenge believed that government policy can affect industrial agglomeration through price index, but agglomeration does not depend entirely on price index but on government policy [5, 6]. Lyons conducted an empirical analysis on the preferential policies implemented by the US state governments to promote industrial agglomeration and found that preferential policies represented by tax incentives and government subsidies have a positive impact on promoting regional enterprise clusters [7]. Cui built a policy analysis framework for makerspace from a three-dimensional perspective of policy tools, value attributes, and life cycle and pointed out that the policy system should pay more attention to the coordination and moderation of policies and suggestions to stimulate market vitality [8]. Xu analyzed 159 makerspace policy documents according to the research ideas of policy type-policy time-regional city by the quantitative research method from the perspective of policy tools [9]. Lei conducted a quantitative analysis on the policies for supporting the development of makerspace introduced by Shanghai by the content analysis method [10]. Wang studied the local policies of Shanghai makerspace from the perspective of dependent multilayer network, formed a supply framework of makerspace, and put forward the policy suggestions on the development of Shanghai makerspace [11]. In recent years, the international academia has recognized that the agglomeration of innovation elements within a certain spatial scope will produce agglomeration externalities and scale effects will promote innovation. More and more scholars at home and abroad have begun to pay attention to the agglomeration of innovative elements, the motivation of the agglomeration of innovative elements, and the relationship between policy guidance and the agglomeration of innovative elements. However, based on the existing literature, while the previous research provides rich experience and reference for this research, there are still certain shortcomings, mainly: the theory of agglomeration of innovative elements is based on the theory of industrial agglomeration, innovation network theory, growth pole theory, and production system. Based on theories and other theoretical foundations, the research on the agglomeration mechanism of innovative elements based on these theories usually ignores the role of government actions in guiding the agglomeration of innovative elements. Although some scholars have pointed out that the role of government intervention in the agglomeration economy has always been a weak link in the current theoretical research on factor agglomeration, few scholars have conducted in-depth research on it. In this paper, an analysis of 17 makerspace development policies of Zhejiang was conducted by NVivo, a qualitative analysis software. By continuously summarizing, 7 core concepts were finally summarized. Through the game evolution method to analyze the mechanism of the internal relationship of innovation factors and build a model of the development factors of the crowd-creation space and through the empirical research on policy performance, it is

found that the balanced distribution of policy attention is not enough, and the government's support for talent policy and service effectiveness is weak.

2. Selection and Statistical Analysis of the Policy Texts

The selected makerspace development policy texts of Zhejiang were all derived from public data from the official websites of Zhejiang Provincial People's Government, vip.chinalawinfo.com, collection, release, and interpretation platform for mass innovation and entrepreneurship policy and other departments. To ensure that the policy sample selection was representative, comprehensive, and accurate, we determined the sample selection principles: first, the relevant policies are searched with "makerspace" and "innovation and entrepreneurship" as the key words; second, the time is selected from March 2015 when the *Guiding Opinions of the General Office of the State Council on Developing Makerspace and Promoting Mass Innovation and Entrepreneurship* was released to December 2019 when the makerspace policy was issued by Zhejiang; third, the policy by Zhejiang Provincial Government and various functional departments is formulated; and fourth, laws and regulations, planning, programs, opinions, methods, notices, announcements, and so on were selected as the policy documents, excluding informal decision-making documents such as approvals, leaders' speeches, and work reports. According to the above principles, 17 makerspace development policies of Zhejiang were finally selected, and the policy documents are shown in Table 1.

3. Construction and Analysis of the Makerspace Development Element Model

3.1. Statistical Analysis of the Policy Texts by Word Frequency. First, the statistical analysis of policy documents was conducted using the "word frequency" function of Nvivo11 (see Figure 1). The overview of makerspace development policies can be intuitively displayed through the word cloud map. Among them, "enterprise," "development," "innovation," "construction," "service," "industry," "technology," "entrepreneurship," and other keywords were in large font size; that is, they were high-frequency words, which showed that innovation and entrepreneurship, enterprise development, science and technology innovation, makerspace construction, service innovation were the key contents of the government in formulating the development policy of makerspace, and they embodied the main contents of makerspace development and involved different stages of innovation and entrepreneurship and related elements. The "enterprise" included small and microenterprises, high-tech enterprises, e-commerce enterprises, and other types of enterprise as well as all aspects of financial support for enterprises, so it was a high-frequency word in the policy text. The "development" was a high-frequency word in the policy text, indicating that the makerspace policy focused on promoting the development of enterprises, industrial development, talent development, and so on. The "innovation," including

TABLE 1: The makerspace development policy documents of Zhejiang.

Table with 3 columns: No., Title, Issuing no. containing 17 rows of policy document details.



FIGURE 1: Word cloud map.

innovation and entrepreneurship, scientific and technological innovation, and service innovation, was the essence of makerspace development, so it was a high-frequency word

in the policy text. The “service” appeared 1,124 times in the policy text, indicating that Zhejiang pays attention to innovation and entrepreneurship services. The “science and

technology” appeared 957 times in the policy text, indicating that scientific and technological innovation and high-tech talents play an important role in the development of makerspace, and the services of competent departments at all levels in scientific and technological innovation, talent training, and talent introduction are extremely important. The cultivation of innovation and entrepreneurial talents and the development of makerspace have always been the focus of economically developed cities, and strengthening service construction is an important way to improve the service quality of makerspace.

Although the word frequency statistic has an intuitive effect, it can only roughly reflect the policy trend. To accurately judge the policy trend, we need to use the “node” function to conduct an in-depth coding analysis of the policy text.

3.2. Data Encoding Analysis and Model Construction

3.2.1. Open Encoding: Acquisition of the Direct Element of Makerspace Development Policy. To better analyze the policy texts, this paper open-encoded the policy documents according to the “localized” [12] principle of Grounded Theory. 30 makerspace development policy texts were imported into Nvivo11 for analysis and coding sentence by sentence. First, the policy documents were encoded into 686 free nodes, which were located at the bottom of the affiliation and were direct elements affecting the makerspace.

3.2.2. Axial Encoding: Construction of the Makerspace Development Element Model. Then, through in-depth analysis according to the relationship between the principal and the subordinate, the 686 free nodes were summarized and formed into 34 second-level nodes. All the contents of the policy texts were encoded as much as possible. The 34 second-level nodes were further summarized and formed 7 first-level nodes: “entrepreneurial incentive,” “fee reduction,” “employment support,” “entrepreneurial environment,” “talent support,” “financial support,” and “service optimization” (see Table 2). So far, after the analysis and node encoding of the 30 collected policy documents, the nodes at all levels of the makerspace development policies of Zhejiang were obtained. Based on the theory of innovation elements [13], the model of makerspace development element was constructed (Figure 2).

3.2.3. Analysis of the Basic Elements of Makerspace Development Policy. The basic idea of evolutionary game is that in a game group, the two parties of the game continue to engage in repeated game activities. Evolutionary game theory is a dynamic game process. The equilibrium point of both parties is experience. When the game process tends to be stable, it will be more realistic in describing the evolution of the decisions of the two parties in the game and the evolution of the system. The basic concept of evolutionary game theory is evolutionary stability strategy, which provides the possibility for evolutionary game theory to study the game process of game groups from the perspective of bounded rationality.

The evolutionary stability strategy reflects the relatively stable situation formed in the dynamic game process. The specific process is to first select some individuals to participate in the game by random selection under a large population. The game strategies of these selected individuals have been specified in advance. After completion, the strategy of some individuals in the group participating in the game has changed; that is, another strategy set beforehand is selected. If the strategy chosen by these individuals does not cause the original strategy of the rest of the individuals to change, then the original strategy has produced a relatively different strategy. With high payouts, individuals who choose mutation strategies will either choose the original strategy to adapt to survival under external pressure or be excluded by the group. Evolutionary game strategy is a stable state in which a population resists the invasion of mutation strategy. Under the incentive of the government’s guiding strategy, the main body of innovation factors chooses whether to enter the crowd-creation space to participate in innovation activities. This can be regarded as a dynamic game process. It has become the optimal strategy for most of the subjects of innovation. At this time, the government’s guiding strategy will play a role in guiding the concentration of innovation. Evolutionary game theory and copying dynamic equations can vividly describe the frequency and frequency of a certain strategy used in a group. This provides theoretical and methodological enlightenment for us to study whether government strategies can be adopted by most innovative elements. It provides a theoretical basis for our understanding of the driving mechanism of the government to guide the agglomeration of innovative elements.

The node value of parent node element of makerspace development policy was 277 (see Table 2). From the reference number of the second-level nodes of policy support, we can see that the financial support had the most nodes, 90, which showed that there were many government support methods. The most important method was financial support, which can support industries and departments, such as financial investment and financial subsidies. The second is employment support. By introducing employment security measures and special subsidies for temporary employment, we can reduce the burden on entrepreneurs. The third was entrepreneurial incentive including various policies established by the government such as innovation incubation funds, entrepreneurial incentive mechanism, and risk compensation, which can improve the operation, management ability, and incubation ability of makerspace. The last was talent support, which was mentioned by only 10 policy documents.

4. Analysis of the Internal Mechanism of Elements for the Government to Guide the Entrepreneurs into the Makerspace

Through the solution to the evolutionary stability strategy [14, 15] between the government and the entrepreneurs, the internal mechanism of elements for the government to guide the entrepreneurs into the makerspace was analyzed.

TABLE 2: Reference points of the elements of makerspace development policy of Zhejiang.

Policy	Element classification	Material source	Reference point
*Makerspace development policy of Zhejiang (267)	Entrepreneurial incentive	16	40
	Fee reduction	13	20
	Employment support	15	64
	Entrepreneurial environment	16	26
	Talent support	10	11
	Financial support	14	90
	Service optimization	11	16

* means the makerspace development policies of Zhejiang came from 17 makerspace development policy documents issued by Zhejiang from 2015 to 2020.

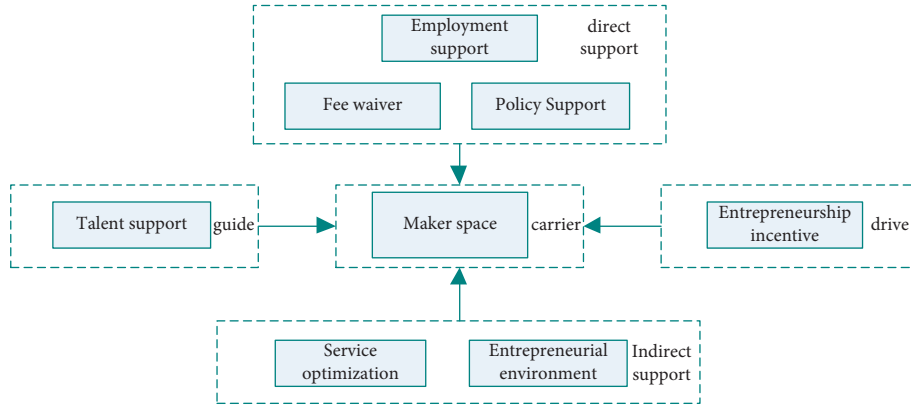


FIGURE 2: The model of makerspace development element.

4.1. Establishment of the Payoff Matrix between the Government and the Entrepreneurs

4.1.1. Basic Assumptions and Parameters of the Game

Assumption 1. Without considering the impact of other external environments, the participants of the game system are composed of two subjects: the government and the maker, and each subject is a bounded rational “economic man.” Make mistakes and choose to adjust your strategy until you make the best decision.

Assumption 2 (opportunity cost of revenue (B) (that is, the basic salary of the current position) [16] and expected entrepreneurial revenue (EX_1)). One of the important reasons [17] why individuals give up their current work to start a business is that they can earn higher revenue than their basic wage (B). Suppose the total revenue realized after the success of the business was I and the tax rate was α , but the premise of the implementation of I was the entrepreneurial success. Suppose the probability (C) of entrepreneurial success was a function of the richness and quality of services provided by the makerspace for the entrepreneurs, i.e., $C = Df(R, U)$. D was the policy effect, which was a function of government subsidies, i.e., $D = D_0h(F_z), h(F_z) > 0$. D_0 was the initial status value. Thus, $C = D_0h(F_z), f(R, U)$. To sum up, $EX_1 = aPI = aID_0h(F_z)$.

Assumption 3 (entrepreneurial investment (S)). Starting a business requires not only energy but also a large amount of basic funds (such as office space costs, industrial and commercial administrative expenses, water-electricity-

Internet charges, and infrastructure purchase fees), as well as part of the entrepreneurial service charges that entrepreneurs enjoy in the makerspace. Here, we regarded all the entrepreneurial investment in the early stage of the business as the entrepreneurial investment (S).

Assumption 4 (entrepreneurial policy incentives (F_c) and opportunity cost of capital (S_{j1})). Entrepreneurs invest the funds (S) obtained through various channels into the startup while abandoning the possible revenue from these funds invested in other areas. Suppose the basic return rate in the capital market was θ and the capital gains tax was η . If these funds were invested into the capital market, the revenue was as follows: $S_{j1} = (1 + \theta)(1 - \eta)S$.

Assumption 5 (Tax (T)). Entrepreneurial success will bring tax (T) to the government [18]. T included the tax paid by the startup ($T_1 = \phi aCI$) and the tax paid by the makerspace ($T_2 = \phi bCI$). To sum up, $T = (\phi a + \phi b)CI$.

Assumption 6 (entrepreneurial effect (Z), incentive derivative cost (S_y), time value (H), entrepreneurial policy incentives ($F = F_c + F_z$)). It included the entrepreneurial policy incentives for the entrepreneurs and the entrepreneurial policy incentives for the makerspace [19].

4.1.2. Construction of Evolutionary Game Revenue Payment Matrix.

Suppose the probability that a maker enters the makerspace to participate in innovation and

entrepreneurship is C_1 , the probability that a maker does not enter the makerspace is $1 - C_1$, the probability that the government chooses to support it is C_2 , and the probability that it does not support is $1 - C_2$. Based on the above model assumptions and parameter settings, the payment matrix of the evolutionary game model between the government and entrepreneurs is shown in Table 3.

4.2. The Construction and Analysis of the Expectation Function of Both Sides of the Game between Government and Entrepreneurs

4.2.1. Building of the Revenue Expectation Function. According to the above analysis, the return expectation function of the entrepreneurs and the government was constructed. Among them,

- (1) The return expectation function of the entrepreneurs was as follows:

$$\begin{aligned} Q_{\text{Enter}} &= C_2[(1 - \phi)aCI + F_c] + (1 - C_2)(1 - \phi)aCI = C_2F_c + (1 - \phi)aCI, \\ Q_{N\text{-Enter}} &= C_2[B + (1 + \theta)(1 - \eta)S] + (1 - C_2)[B + (1 + \theta)(1 - \eta)S] = B + (1 + \theta)(1 - \eta)S, \\ Q_E &= C_1Q_{\text{Enter}} + (1 - C_1)Q_{N\text{-Enter}}. \end{aligned} \quad (1)$$

- (2) The return expectation function of the government was as follows:

$$\begin{aligned} Q_{\text{INSPIRE}} &= C_1(T + Z - F - S_y + H) + (1 - C_1)[\phi B + \eta(1 + \theta)S - S_y], \\ Q_{N\text{-INSPIRE}} &= C_1(T + Z - H) + (1 - C_1)[\phi B + \eta(1 + \theta)S], \\ Q_G &= C_2Q_{\text{INSPIRE}} + (1 - C_2)Q_{N\text{-INSPIRE}}. \end{aligned} \quad (2)$$

4.3. Solution to the Evolutionary Stability Strategy Based on Replication Dynamic Equation

- (1) The replicated dynamic equation of the entrepreneurs was as follows:

$$\begin{aligned} M(C_1) &= \frac{dC_1}{dt} = C_1(Q_{\text{ENTER}} - Q_E) = C_1(1 - C_1)\{[C_2F_c + (1 - \phi)aCI] - [B + (1 + \theta)(1 - \eta)S]\}, \\ M'(C_1) &= (1 - 2C_1)\{[C_2F_c + (1 - \phi)aCI] - [B + (1 + \theta)(1 - \eta)S]\}. \end{aligned} \quad (3)$$

Making $M'(C_1) = (dC_1/dt) = 0$, we obtained $C_1^* = 0$, $C_1^* = 1$, and $C_2^* = ((B + (1 + \theta)(1 - \eta)S - (1 - \phi)aCI)/(F_c))$.

Judging from the stability theorem of the replicated dynamic differential equation and the evolutionary stability strategy, when $M(C_1) = 0$, $M'(C_1) < 0$, P^* was the evolutionary stability strategy.

Discussion. If $C_2 = ((B + (1 + \theta)(1 - \eta)S - (1 - \phi)aCI)/(F_c))$, then $M(C_1) = 0$, $M'(C_1) = 0$; that is, all y -axis levels were stationary; when the government incentives reached $C_2 = ((B + (1 + \theta)(1 - \eta)S - (1 - \phi)aCI)/(F_c))$, the possibility of entrepreneurs staying in the makerspace was all stable.

If $C_2 > ((B + (1 + \theta)(1 - \eta)S - (1 - \phi)aCI)/(F_c))$, for $C_1^* = 0$ and $C_1^* = 1$, $M'(0) > 0$ and $M'(1) < 0$. Now, $P_1^* = 1$ was the unique evolutionary stability strategy; when the government incentives reached to

a certain extent and continued to increase, it was the best choice for entrepreneurs to move in the makerspace.

If $C_2 < ((B + (1 + \theta)(1 - \eta)S - (1 - \phi)aCI)/(F_c))$, for $C_1^* = 0$ and $C_1^* = 1$, $M'(0) < 0$ and $M'(1) > 0$. Now, $P_1^* = 0$ was the unique evolutionary stability strategy; when the government incentives were weak enough and continued to weaken, the possibility of entrepreneurs staying in the makerspace would gradually reduce. At this time, it was the best choice for entrepreneurs not to start a business. The relationship between the intensity of government incentives and the possibility of capital flowing into the crowd-creation space to participate in innovation and entrepreneurship is shown in Figure 3.

- (2) The replicated dynamic equation of the government was as follows:

TABLE 3: The payoff function matrix of the entrepreneurs and the government under different strategies.

Government policy	Entrepreneur	
	Entry C_1	No entry $1 - C_1$
C_2	$K_E = (1 - \phi)aCI + F_c$ $K_G = T + Z - F - S_y + H$	$K_E = B + (1 + \theta)(1 - \eta)S$ $K_G = \phi B + \eta(1 + \theta)S - S_y$
$1 - C_2$	$K_E = (1 - \phi)aCI$ $K_G = T + Z - H$	$K_E = B + (1 + \theta)(1 - \eta)S$ $K_G = \phi B + \eta(1 + \theta)S$

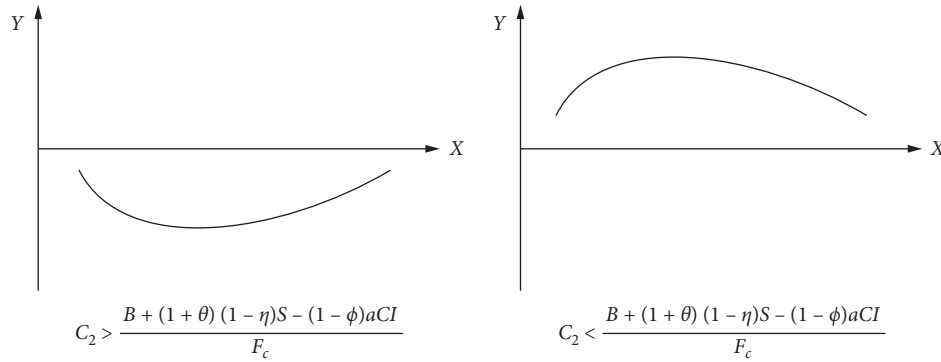


FIGURE 3: The replicated dynamic equation phase diagram for the possibility of entrepreneurs staying in the makerspace.

$$M(C_2) = \frac{dC_2}{d} = C_2(Q_{INSPIRE} - Q_G) = C_2(1 - C_2)[C_1(2T - F) - S_y], \tag{4}$$

$$M'(C_2) = (1 - 2C_2)[C_1(2T - F) - S_y].$$

Making $M(C_2) = (dC_2/d) = 0$, we obtained $C_2^* = 0, C_2^* = 1$, and $C_1^* = ((S_y)/(2T - F))$.

Judging from the stability theorem of the replicated dynamic differential equation and the evolutionary stability strategy, when $M(C_2)=0$ and $M'(C_2) < 0$, C^* was the evolutionary stability strategy.

Discussion. When $C_1 = (S_y/(2T - F))$, then $M(C_2) = 0$ and $M'(C_2) = 0$; that is, all y -axis levels were stationary; when the government incentives reached $C_1 = (S_y/(2T - F))$, the government incentives were all stable.

If $C_1 > (S_y/(2T - F))$, for $C_2^* = 0$ and $C_2^* = 1$, $M(0) > 0$ and $M'(1) < 0$. Now, $C_2^* = 0$ was the unique evolutionary stability strategy; when the possibility of entrepreneurs staying in the makerspace was strong enough and continued to increase, the government incentives would increase. At this time, it was the best choice for the government to increase incentives.

If $C_1 = (S_y/(2T - F))$, for $C_2^* = 0$ and $C_2^* = 1$, $M(0) > 0$ and $M'(1) < 0$. Now, $C_2^* = 0$ was the unique evolutionary stability strategy; when the possibility of entrepreneurs staying in the makerspace was weak enough and continued to decrease, the government incentives would decrease. At this time, it was the best choice for the government to decrease incentives.

The dynamic trend and stability of the government incentives are shown in Figure 4.

4.4. Analysis of the Equilibrium Solution of Game. Based on the above analysis, the replicated dynamic relationship between the government and the entrepreneurs was constructed, which was represented by a two-dimensional plane coordinate, as shown in Figure 5.

In Figure 5, $O(0, 0)$ and $B(1, 1)$ were evolutionary stability strategies. $O(0, 0)$ meant that entrepreneurs did not move in the makerspace, and the government did not take financial support measures. $B(1, 1)$ meant that entrepreneurs actively moved in the makerspace to start businesses, and the government took vigorous financial support measures. In the upper right area of Figure 5, the actions of the both sides converged to $B(1, 1)$, where both sides achieved a Pareto equilibrium [20]. In other words, there was no better point (except $B(1, 1)$) in the above plane area which can make at least one side become better without making the existing situation worse.

In the lower left area of Figure 5, the actions of the both sides converged to $O(0, 0)$, where the both sides achieved a Pareto equilibrium. In other words, there was no worse point (except $O(0, 0)$) in the above plane area which can make at least one side become worse without making the existing situation better. In other areas of Figure 5, the convergence direction of the actions of the both sides would be uncertain.

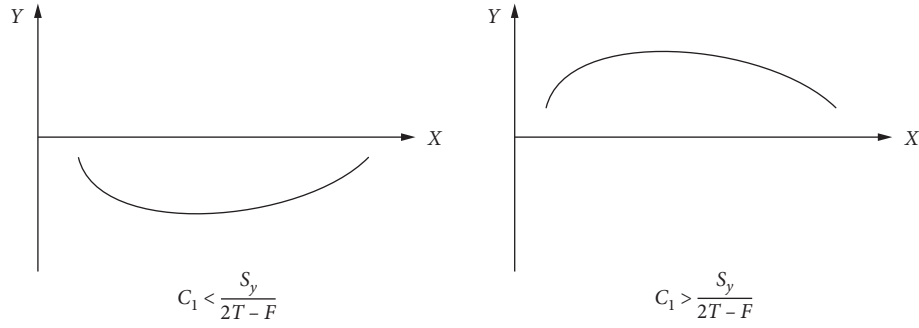


FIGURE 4: The replicated dynamic phase diagram of the government incentives.

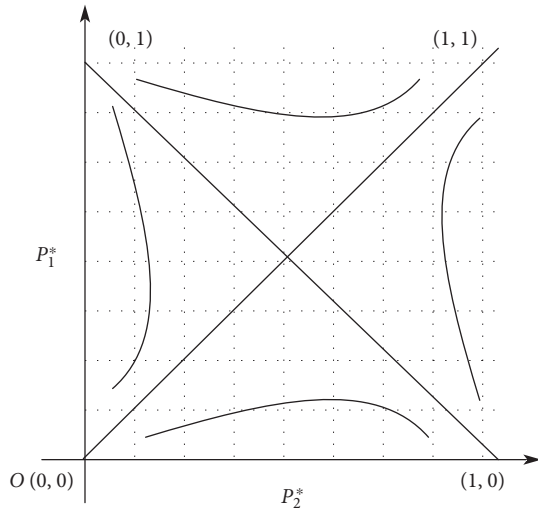


FIGURE 5: The schematic diagram of the replicated dynamics and stability of the both sides.

Through further analysis, if the system was made to converge to the optimal Pareto equilibrium point $B(1, 1)$ with high probability, the equilibrium point of the both sides should appear in the upper right area of Figure 5. Now, the following conditions should be met: $C_1 > (S_y / (2T - F))$ and $C_2 > ((B + (1 + \theta)(1 - \eta)S - (1 - \phi)aCI) / (F_c))$. Based on this, we obtained the following.

In $C_2 = ((B + (1 + \theta)(1 - \eta)S - (1 - \phi)aCI) / (F_c))$, there was a positive relationship between the basic wage (B), capital market value (r), entrepreneurial investment (S), individual income tax rate (ϕ), and C_2^* ; there was a reverse relationship between the capital gains tax (η), the proportion of income distribution of innovation achievements (a), the success rate of entrepreneurship (C), the total income of innovation achievements (I), and C_2^* . Through the above analysis, we got $C = Df(R, U) = D_0g(F_z)f(R, U)$. Therefore, it was further concluded that there was also a reverse relationship between the quantity R and quality U of the entrepreneurial service types provided by the makerspace, the financial subsidies provided by the government to the makerspace, and C_2^* .

In $C_1^* = (S_y / (2T - F))$, there was a positive relationship between the government incentives $F = F_c + F_z$ (F_c means government incentives for the entrepreneurs and F_z means government incentives for the makerspace), incentive

derivative cost (S_y), and C_1^* ; there was a reverse relationship between the time value (M) and C_1^* .

5. Empirical Analysis

5.1. Data Acquisition and Validity Analysis

- (1) *Data Acquisition.* Same as the data acquired in the index system, the data on the implementation of policy tools also came from 52 makerspaces of Zhejiang, but they were acquired through the satisfaction survey of the policies formulated. The policies were tested by Likert five-level scale [21], with the policies not enjoyed by the makerspace being 0 point. The policies enjoyed by the makerspace were measured by satisfaction, which was divided into five dimensions: very satisfied, satisfied, general, unsatisfied, and very unsatisfied, with the scores of 5–1 point, respectively. Although 52 makerspaces were surveyed, 38 questionnaires were recovered, including 7 invalid questionnaires and 31 valid questionnaires.
- (2) *Reliability and Validity Test.* The reliability of the support policy scale was analyzed using SPSS, a statistical analysis software, to test its internal consistency, namely, to measure whether each policy reflected the effect that the policy should be expressed.

Reliability index, also known as reliability coefficient, is a quantitative method to test reliability with Cronbach's ϕ . Cronbach's α of the policy scale was 0.865, indicating that the internal reliability of the policy scale was good.

Validity is to measure whether the comprehensive evaluation system can accurately reflect the evaluation purpose and requirements and refers to the correctness of the characteristics measured by measuring tools. The higher the validity, the higher the correctness of the characteristics measured by measuring tools. The author classifies the policies according to the dimensions of employment support policies, entrepreneurship incentive policies, talent support policies, and financial support policies and designs the items of the scale according to the policies, as shown in Table 4. By deleting and modifying some items of the scale, we get the optimized efficiency scale, as shown in Table 5.

TABLE 4: Government policy elements scale of makerspace.

Variable dimensions	Measurement questions
Employment support policies	(Aa1) Introduce employment protection measures
	(Aa2) Temporary special subsidy for employment
	(Aa3) Provide employment guidance services
Entrepreneurial incentive policies	(Ab1) Award based on the crowd-creation space recognized by the city, province, country, and other different levels
	(Ab2) Establish various venture capital funds
	(Ab3) Support researchers from universities, scientific research institutes, and corporate executives to retain their jobs and start their own businesses
Talent support policies	(Ac1) Establish entrepreneurship college, innovation and entrepreneurship resource construction, and entrepreneurship practice teaching
	(Ac2) Provide support policies for the entry and exit of talents, housing, settlement, children’s enrollment, and employment of family members
	(Ac3) Expand the scale of talent introduction
Financial support policies	(Ad1) Reward based on the number of incubating companies, the number of successfully incubated companies, and the projects introduced
	(Ad2) Subsidies for rent, broadband fees, water and electricity, purchase of facilities, and so on
	(Ad3) Provide loan interest discounts and guarantee fee subsidies for startups in the space

TABLE 5: Cronbach’s alpha of innovation elements in each dimension.

Dimension	Item	CITC	Cronbach
Employment support policies	Aa1	0.536	0.865
	Aa2		0.850
	Aa3		0.864
Entrepreneurial incentive policies	Ab1	0.665	0.875
	Ab2		0.864
	Ab3		0.855
Talent support policies	Ac1	0.428	0.862
	Ac2		0.870
	Ac3		0.845
Financial support policies	Ad1	0.502	0.844
	Ad2		0.865
	Ad3		0.865

5.2. Construction of the Policy Effect Measurement Model.

In this study, the general multiple regression analysis model was constructed based on the statistical analysis of data through survey questionnaires. The policy category was n ; $y_1, y_2, y_3,$ and y_4 were the explanatory variables, which represented employment support policies, entrepreneurial incentive policies, talent support policies, and financial support policies, respectively; h was the intercept item, which was a constant; t was the number of statistical dimensions of the model data; ZK_0 was the innovation support ability of makerspace; ZK_1 was the maker attraction ability of makerspace; ZK_2 was the entrepreneurial resource guidance ability of makerspace; and ZK_3 was the entrepreneurial output ability of makerspace; the model was as follows:

$$ZK_t = a_i y_i + \dots + a_n y_n + h. \tag{11}$$

5.3. Empirical Results. The SPSS was used to bring the processed raw data into the model for regression, as shown in Table 6.

The regression results of ZK_0 showed that employment support policies, entrepreneurial incentive policies, and financial support policies had a positive effect on the ZK_0 , while talent support policies had a negative effect on ZK_0 . The regression results of ZK_1 showed that employment support policies, entrepreneurial incentive policies, and financial support policies had a positive effect on ZK_1 , while talent support policies had a negative effect on ZK_1 . The effect of policy on ZK_1 was consistent with that on ZK_0 . The difference was that talent support policies had an insignificant negative effect on ZK_1 and entrepreneurial incentive policies had the strongest positive effect on ZK_1 , followed by financial support policies. The regression results of ZK_2 showed that only financial support policies had a positive effect on ZK_2 . The regression results of ZK_3 showed that employment support policies and entrepreneurial incentive policies had a positive effect on ZK_3 , while talent support policies and financial support policies had a negative effect on ZK_3 .

6. Problem Analysis and Policy Improvement Suggestions

6.1. Insufficient Balanced Distribution of Policy Attention and Weak Government Support for Talent Support Policies and Service Support Effectiveness. Judging from the material source and reference point of the parent node of Zhejiang, there was a large gap between the times of nodes mentioned in the policy documents, and the government focused on financial support, entrepreneurial environment, entrepreneurial incentives, and employment support. As for the agglomeration ability of entrepreneurs, the influence coefficient of employment support policies, entrepreneurial incentive policies, and financial support policies was positive, which suggested that the three types of policies broke through the policy trigger point (equilibrium point) in the process of implementation and realized the agglomeration of entrepreneurs in the makerspace. The influence coefficient of

TABLE 6: Empirical results.

	Policy performance			
	Innovation support ability	Attraction ability	Resource guidance ability	Output ability
Employment support policies	0.0578	-0.504**	0.171*	0.148*
Entrepreneurial incentive policies	-0.239*	-0.139*	0.067	0.128*
Talent support policies	-0.277*	0.279*	-0.401*	-0.427***
Financial support policies	0.278*	-0.145*	-0.022	0.08*

Note. The figure in brackets is t ; *, **, and *** indicate significance at the significance level of 10%, 5%, and 1%, respectively.

the three types of policies in order from large to small was financial support policies > entrepreneurial incentive policies > employment support policies. It can be seen that in the process of entrepreneurs to decide whether to enter the makerspace, the financial support policies were their first consideration, followed by the entrepreneurial incentive policies. Although the employment support policies broke through the policy trigger point (equilibrium point), the effect was not significant. The talent support policies were the opportunity element of the development of makerspace and the core element of innovation. The effect of talent service was the motivation goal of the development of makerspace, and the powerful talent policy was an important force to promote the development of makerspace. However, at present, Zhejiang does not pay enough attention to the makerspace policies in talent and ignores the support for opportunities and motivation goals when formulating policies. Therefore, the formulation of makerspace development policies must strengthen the emphasis on talent support policies and service results.

6.2. Policy Improvement Suggestions. The report to the 19th National Congress of the Communist Party of China calls for implementing a more active, more developed, and more effective talent policy to gather talents from all over the world and accelerate the building of a talent power. Zhejiang Provincial Talent Work Conference put forward to highlight the “best and brightest talents” guidance and to build the best province of talent ecology. Therefore, the implementation of accurate introduction of talents is imperative. The accurate introduction of talents is to introduce and manage talents to meet the needs of makerspace industry and ensure that everyone can become talented and everyone can show their talents. The state and local governments have also issued a series of policy documents on talent introduction and management, such as “Thousand Talents Program,” “Hundred Talents Program,” and “151 Talents.” However, studies have shown that the local talent introduction policies have a high homogenization and the lack of differentiation, leading to the single talent introduction channel and the homogenization of talent [22]. Innovation drives development, and talent leads innovation. To gain influence for makerspace in the competition, the key is to accelerate the agglomeration of makerspace innovative talents. Therefore, we should pay attention to talent planning and make Zhejiang become the place of talent agglomeration by means of the talent introduction model and talent flow guarantee mechanism. First, the talent introduction model is innovated, and equal emphasis on talent introduction

and talent cultivation is advocated. The traditional “government-led” and “capitalized” talent introduction model is reformed [23], and the talent introduction model of “combining government support and market mechanism” is established. In other words, the government plays a guiding and promoting role in the macroaspects such as the reward policy and guarantee policy of talent introduction. And the employers implement other microaspects such as talent introduction way, talent types, standards, treatment, scientific research, and entrepreneurial working environment in accordance with the market mechanism. According to the needs of industrial development, the employers should do a good job in top-level design and overall planning, clear the goal of talent introduction, attract talents as needed, and focus on the cultivation of makerspace talents. According to the needs of local economic development, the employers should formulate the talent cultivation mechanism and strengthen the follow-up cultivation of local talents and introduced talents so that the introduced talents can drive local talents, and they learn from each other, creating an environment for harmonious development.

In addition, attention is paid to both the introduction of talents at home and abroad and the guarantee mechanism of talent loss. By 2050, the total number of international immigrants will reach 405 million, with about 70% of skilled immigrants [24, 25]. Attracting and retaining these talents are the source of impetus for the future development of makerspace. Therefore, talents are introduced according to the regional makerspace development practice and the talent policies are dynamically adjusted for different development stages of makerspace, to ensure the organic integration of talent development with makerspace and social and economic development as well as the precise docking of talent introduction with makerspace industry demand and to reduce the waste of resources and talent loss caused by blind introduction. Besides, pay close attention to the innovation and entrepreneurship work after the introduction of talents to make them adapt to the innovation and entrepreneurship atmosphere of makerspace as soon as possible, track, analyze, evaluate and feedback the innovation and entrepreneurial behavior and achievements so as to timely improve the talent management mechanism, adjust the talent strategy, and ensure the stable development of the talent team.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

The research presented in this paper was supported by Zhejiang Provincial Soft Science Research Plan Project of China “Research on the Driving Mechanism and Performance Evaluation of Policy Guiding Innovation Factors to Gather in Makerspace: An Empirical Study of Zhejiang” (grant no. 2019C35050).

References

- [1] X. Z. Xu, “Construction of localization theory based on empirical research—comment on research on localization of educational policy process in China,” *Research in Educational Development*, vol. 37, no. 5, pp. 83–84, 2017.
- [2] M. Porter, *The Competitive Advantage of Nations*, Free Press, New York, NY, USA, 1990.
- [3] J. Jacobs, *The Economy of Cities*, Random House, New York, NY, USA, 1969.
- [4] J. Jacobs, *The Death and Life of Great American Cities*, Random House, New York, NY, USA, 1961.
- [5] R. Baldwin, R. Forslid, P. Martin, G. Ottaviano, F. R. Nicoud, and F. Robert-Nicoud, *Economic Geography and Public Policy*, Princeton University Press, Princeton, NJ, USA, 2003.
- [6] R. Fenge, M. von Ehrlich, and M. Wrede, “Public input competition and agglomeration,” *Regional Science and Urban Economics*, vol. 39, no. 5, pp. 621–631, 2009.
- [7] T. S. Lyons, *Economy Development: Strategies for State and Local Practice*, International City/County Management Association (ICMA), Washington, DC, USA, 2010.
- [8] X. M. Cui, “Text analysis of mass innovation space policy based on three-dimensional perspective,” *Science and Technology Management Research*, vol. 37, no. 17, pp. 30–36, 2019.
- [9] S. B. Xu, “Policy research on the development of mass maker space from the perspective of policy tools,” *China Science and Technology Forum*, vol. 35, no. 6, pp. 29–39, 2019.
- [10] L. H. Lei and T. M. Jia, “Research on maker-space policy system of Shanghai,” *Shanghai Economic Research*, vol. 37, no. 3, pp. 34–41, 2017.
- [11] H. H. Wang, Y. Li, L. J. Xiong, M. Du, and Q. Sun, “Local policy supply of mass innovation space from the perspective of interdependent multi-layer network: a case study of Shanghai,” *Research and Development Management*, vol. 31, no. 6, pp. 13–23, 2019.
- [12] J. M. Corbin and A. L. Strauss, *Fundamentals of Qualitative Research: Procedures and Methods for Forming Grounded-Theory*, Chongqing University Press, Chongqing, China, 3rd edition, 2015.
- [13] G. S. Chen, “A review of the theoretical model of the core factors of entrepreneurship,” *Advances in Domain*, vol. 17, no. 12, 2007.
- [14] Z. B. Li, S. Liu, and L. L. Liu, “Research on the value added path of incubating small and medium-sized enterprises by crowd maker space: experience from 69 crowd maker spaces in Tianjin,” *Science and Technology Progress and Countermeasures*, vol. 35, no. 3, pp. 72–79, 2018.
- [15] B. W. Zhou and Z. S. Zhang, “Crowd innovation culture: an urgent field to be studied,” *Management Modernization*, vol. 4, 2017.
- [16] F. Huang and L. Q. Liu, “Maker space culture with Chinese characteristics,” *Learning and Practice*, vol. 8, 2017.
- [17] L. Zhao and G. R. Chai, “Research on the development and evolution of enterprise and maker space innovation cooperation network,” *Studies in Science of Science*, vol. 35, no. 2, 2017.
- [18] Y. P. Li and W. Chen, “Research on the ecological network elements and capacity generation of maker-oriented platform organization,” *Economic Management*, vol. 6, 2017.
- [19] People’s Daily Online, “Comment on: avoid the homogeneity of the gen space supporting policies need to be born,” 2015, <http://scitech.people.com.cn/n/2015/1105/c1007-27782036.html>.
- [20] Y. S. Xi, “The allocation of local governments’ attention in the construction of double first class: an analysis of NVIVO software based on 30 provincial policy text,” *Research on Education Development*, vol. 37, no. 21, pp. 31–38, 2017.
- [21] J. L. Yu and Q. C. Yao, “From data to evidence: research on the construction of government evidence-based decision-making mechanism in the era of big data,” *Chinese Administrative Management*, vol. 27, no. 4, pp. 81–87, 2020.
- [22] S. Wu, “Study on the homogeneity of China’s overseas talent promotion policy from the perspective of governmental relations,” *Chinese Administration Management*, vol. 21, no. 9, pp. 89–92, 2014.
- [23] Y. Li and J. P. Zhang, “Introduction and management of high-level talents in the “Double First-class” construction,” *Science and Technology of Chinese Universities*, vol. 25, no. 12, 2019.
- [24] J. Wu, “Strengthening the cornerstone of the modernization of national governance with talents,” *Guangming Daily, Amoxicillin*, vol. 8, no. 7, 2019.
- [25] J. Q. Chen, A. T. Xu, and J. C. Li, “The continuous distance measurement of crowd maker space agglomeration and its influencing factors,” *Business Economics and Management*, vol. 39, no. 3, pp. 89–97, 2019.

Research Article

DAuGAN: An Approach for Augmenting Time Series Imbalanced Datasets via Latent Space Sampling Using Adversarial Techniques

Andrei Bratu  and Gabriela Czibula 

Faculty of Mathematics and Computer Science Babeş-Bolyai University, Cluj-Napoca, Romania

Correspondence should be addressed to Andrei Bratu; bratuandrei0@gmail.com

Received 7 July 2021; Accepted 14 September 2021; Published 12 October 2021

Academic Editor: Sze-Teng Liong

Copyright © 2021 Andrei Bratu and Gabriela Czibula. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Data augmentation is a commonly used technique in data science for improving the robustness and performance of machine learning models. The purpose of the paper is to study the feasibility of generating synthetic data points of temporal nature towards this end. A general approach named DAuGAN (Data Augmentation using Generative Adversarial Networks) is presented for identifying poorly represented sections of a time series, studying the synthesis and integration of new data points, and performance improvement on a benchmark machine learning model. The problem is studied and applied in the domain of algorithmic trading, whose constraints are presented and taken into consideration. The experimental results highlight an improvement in performance on a benchmark reinforcement learning agent trained on a dataset enhanced with DAuGAN to trade a financial instrument.

1. Introduction

Data augmentation is a vast and often used method for enhancing the amount of data available for training a *machine learning* (ML) model. It is well known that the amount and quality of data available are closely bounded to the success of any ML project, independent of the application domain. There are multiple data augmentation procedures, which are often specific to the application domain and the specific dataset that is used.

For example, image-based machine learning tasks often employ operations of contrast adjustment, flipping, translation, cropping, rotation, color space manipulation, etc. These present new contexts to the model, helping it to better generalize and to avoid overfitting [1]. Another example of data augmentation is the SMOTE, or the Synthetic Minority Oversampling Technique, whose purpose is to alter the dataset presented to the algorithm by presenting minority-class data points to the classifier more often than they naturally occur (oversampling), while minimising the rate at which the majority class appears (undersampling). This increases the sensibility of the model for the subrepresented

data class and has applications such as identifying fraud credit card transactions [2].

A more recent augmentation method involves using a *generative adversarial neural network* (GAN) architecture, whose ability to reproduce a statistical distribution is repurposed for creating new, convincing examples of a poorly represented class, or generally any point of the dataset [3]. GANs are an important machine learning paradigm. Two neural networks engage in a zero-sum game where the generator network attempts to generate new samples, while the discriminator discerns between real samples and generated samples. The end goal is to train the generator into reproducing the initial train distribution as close as possible. GANs have been used to great effect, with examples such as reproducing the effects of dark matter on astronomical observations [4], generating photorealistic human faces [5], or applying style transfer operations in the audiodomain [6].

Identifying imbalanced classes and enhancing their presence in the time series would not be devoid of practical applications. One such example is *securities trading*. Securities are defined as any financial instrument that can be bought or sold via an accredited intermediary, creating a

supply and demand market. An example is the stock market, which allows buying and selling “shares”: discreet units of ownership in a company. While the price of any share has a correlation with the business performance, there is research that indicates the market’s sentiment and domain-specific factors such as national interest rate create a cyclical effect on the price evolution [7, 8].

The current paper is based on work originating from two research questions:

- (1) RQ1. Is it possible to improve the performance of *reinforcement learning*- (RL-) based trading algorithms through augmenting training data using adversarial techniques?
- (2) RQ2. What is the performance gain of the RL agent trained on the enhanced data over a baseline RL agent trained on the initial data, without augmentation?

To this end, a general approach named *Data Augmentation using GANs* is proposed for identifying poorly represented sections of security-related time series. Leveraging that autoencoder neural network architectures can encode and decode complex temporal dependencies to and from a latent space, the proposal reduces the problem of identifying poorly represented time series periods into a clustering problem [9] and synthesises new examples of the minority class using a GAN architecture. A method of integrating synthesised data points into the original time series, respecting the original constraints of the data, is presented, and experiments are carried out for measuring the performance improvement on benchmark reinforcement learning algorithms.

Parallels between the proposed method and the TimeGAN method proposed by Yoon et al. for generating arbitrary time-series [10] are acknowledged. The proposed method improves by particularising the problem to the constraints of security-related time series on one hand and the problem of data series minority data augmentation on the other one.

Section 2 introduces fundamental concepts used by the approach, along with a literature review on time series generation. DAuGAN is introduced in Section 3 along with proposed methodology, whilst experimental results and their analysis are presented in Section 4. The conclusions of the paper and directions to further improve and extend DAuGAN are outlined in Section 5.

2. Background

This section presents a literature overview of the technical notions used in this paper. The importance and evolution of generative adversarial and autoencoder networks are presented, together with a brief review on reinforcement learning. Historical approaches related to data augmentation and data synthesis on time series are also presented.

2.1. Generative Adversarial Networks. GANs is a deep learning architecture that have been first introduced by Ian Goodfellow et al. [11], which has been heavily used in image-

based tasks, from synthesising new images [12] to reproducing the content of one image in the style of another.

At a very high level, the generative adversarial network technique pits two deep neural networks against each other in a zero-sum game. One of the networks, the *generator*, acts as a map from a latent space towards a desired distribution, sampling noise from the latent space that is synthesised as closely as possible to a point in the distribution. Its counterpart, the *critic*, is fed samples from both the real distribution and from the generator, with the goal of deciding whether the sample is “real” or “fake.” Over time, the two networks improve at their goal, resulting in not only better fakes from the generator but also a better ability to discern the fakes from the critic. Ideally, the system converges towards equilibrium where the critic can no longer separate between the two classes; i.e., it assign equal probability for any sample to be either one of the classes.

Formally, the generator attempts to minimise the value of the following loss function, while the discriminator attempts to maximise it: $E_x [\log(D(x))] + E_z [\log(1 - D(G(z)))]$, where x is a random variable corresponding to the real distribution, z is a random variable assigned to the generated distribution, $G(x)$ is the generator’s output, $D(x)$ is the discriminator’s output, and E_v denotes the expected value over all instances v .

The expected value is used to indicate that the loss is the average over all samples of the batch at a given training step. The critic assigns values from 0 to 1, estimating the probability that a sample is real. Letting x denote the real distribution and z denote the synthesised distribution, the critic attempts to maximise this loss, the upper bound being obtained when all labels are properly assigned, while the generator attempts to minimise it by controlling the second term, i.e., generating more convincing examples, signified by the $G(z)$ term. The log operations are derived from the cross entropy between the real and fake distributions.

Notorious problems affecting GANs are *mode collapse* and *vanishing gradients*. *Vanishing gradients* is a general issue in machine learning, where gradients prove insufficient for the machine learning model to update meaningfully, while this issue has classically occurred in neural models with high depth [13] or recurrent networks such as long-short term memory architecture [14]. However, the issue manifests particularly in the case of generative adversarial networks: the unadjusted loss formulation presented above will result in a critic that converges faster than the generator. Thus, the critic cannot offer constructive feedback for the generator to improve on, since it perfectly discerns between real and fake.

A connected issue with vanishing gradients that is faced by generative adversarial networks is *mode collapse*. The problem manifests on the generator’s side by mapping all latent points to the same synthesised sample. From the perspective of game theory, both *mode collapse* and *vanishing gradients* issues are caused by the two players, the critic and the generator, converging to a local, undesired optimum of the game that does not offer enough incentives for any of the networks to update their weights [15].

Several improvements on the domain transfer sub-problem have been addressing the mode collapse issue. CycleGAN [16] introduces the following adjustments: instead of sampling from a latent random space, the generator samples from the input space of the input domain, with output in the target domain. The critic is fed both generated images and those belonging to the target domain, thus encouraging the generator to learn a better mapping between input and target. Thus, the improvement resides in translating the task into an unsupervised task, as the generator is ideally able to map any image from the first domain to an image in the target domain. A limitation of the CycleGAN paper is the domains being required to be homogeneous [17]. An improvement over the CycleGAN is represented by TraVeLGAN [18], which adds to the classical two network architectures formed of a generator and critic, a third, siamese network, and eliminates the domain homogeneity constraint [19].

The Wasserstein variation of the generative adversarial network architecture (WGAN), authored by Arjovsky et al. [20] offers a robust method to train GAN architectures. The WGAN improves the stability of learning, eliminates problems such as mode collapse, and provides meaningful learning curves useful for hyperparameter searches.

2.2. Autoencoders. The autoencoder (AE) architecture uses a two-part neural network to transform the input in a compressed and meaningful representation using the encoder part, recreating the input using the decoder part [21]. The technique proves to be immensely flexible and is of interest to the purpose of this thesis since former research proved that autoencoders are able to encode and decode complex temporal features into the latent space [9]. Furthermore, there are no special theoretical aspects to be considered over a general-purpose deep learning architecture. The autoencoder is presented as two symmetrical parts, with a small, latent representation in the middle, usually trained to minimise the mean squared error between the distribution and itself.

AEs can be interpreted as an improvement over the statistical technique of principal component analysis. Principal component analysis with p dimensions identifies an orthonormal base of p vectors that best identify the variance of the input distribution [22]. This technique is limited to linear representations, unlike the manifold organized by the autoencoder. Thus, the AE is able to construct higher fidelity correlations between the original and latent spaces, preserving relationships. Furthermore, there are multiple accounts in the literature in using the latent representation over the initial dataset with great effect for increased classification performance, better interpretability of the obtained clusters, or better ability to generalize over the latent representation [23–25].

Salakhutidnov and Hinton restrict the latent representation to a binary code which is interpreted as the output of a black-box hash function modelled by the encoder. The hashing is applied in the field of document retrieval, where the hashing of the query is used to retrieve the directly associated documents plus documents from neighbouring

hashes. The task has also been approached from a generative approach by Hansen et al. in *Unsupervised Neural Generative Semantic Hashing* [26].

2.3. Reinforcement Learning. Reinforcement learning (RL) is a paradigm of the machine learning field where problems are modelled around two fundamental notions: agents and environments. Agents are able to interact with the environment via a defined set of “actions” which change the environment’s “state.” Defining the problem’s solution as a desirable environment state, the agent is conditioned via “rewards” and “punishments” to reach this favourable state. RL purpose is to teach an agent the optimal policy of acting inside an environment. The environment can at any moment be in a certain *state* that can be changed by the agent’s *actions*. The agent receives feedback from the environment in the form of a *reward*. Using a tradeoff between reward and value (future reward received by the agent by taking a certain action in a particular state), the agent learns a policy that decides the best course of action for a given state.

This flexible framework has allowed to model complex real-world situations: scheduling drug administration to patients with chronic conditions in order to minimise risk of negative interactions [27, 28], minimising energy costs associated with cooling of data centers [29], or outmatching human players in games such as Go [30].

RL is facing a growing interest in the discipline of algorithmic trading [31, 32]. The interest can be explained by the ease with which the problem can be modelled: given the price fluctuation of a certain instrument, an agent’s purpose is to maximise the overall profit. Current frontier in reinforcement learning focuses on improved training performance, particularly incentivising the agent to explore multiple action courses and breaking the state causality effect on training by sampling and replaying random past states [33]. Intuitively, these improvements focus on offering the agent the ability to retrospect and decide on past better courses of action.

2.4. Time Series Generation. GAN-based methods or generative adversarial network models have emerged as a popular technique for generating or augmenting datasets, especially with images and videos. However, GANs give poor fidelity in networking data, which has both complex temporal correlations and mixed discrete-continuous data types. Although GAN-based time series generation exists—for instance for medical time series—such techniques fail on more complex data exhibiting poor autocorrelation scores on long sequences while prone to mode collapse.

TimeGAN architecture introduced by Yoon et al. [10] is of strong interest for the proposed method, as it reinforces the idea that latent spaces can be used to better understand the original time series distribution of the data. Specifically, the paper proposes using two latent spaces, H_S and H_X , where S represents the mathematical space of static features of the time series, e.g., gender, while X represents the space of temporal dependencies of the time series, e.g., the cholesterol level as the person ages. The paper asserts that

instead of using only a generator-discriminator system for creating new samples, introducing supervised learning to the unsupervised generation will increase the fidelity of generated data. The supervised loss comes from two encoder-decoder pairs ($h: S \rightarrow H_S, e: H_S \rightarrow S$) and ($h_X: X \rightarrow H_X, e_X: H_X \rightarrow X$) between the initial space and latent space, with the GAN networks learning to directly replicate the latent vectors: ($g: Z_S \rightarrow H_S, d: H_S \rightarrow \text{IR}$) and ($g_X: Z_S \rightarrow H_S, d: H_S \rightarrow \text{IR}$), where Z_S and Z_X are the space the random noise is sampled from. Of particular interest is the use of recurrent neural networks throughout the architecture. Notably, g_X features the use of a 2-step autoregression dependency for creating the temporal latent vector. Recurrent neural networks are also used for encoding and decoding between X and H_X .

DoppelGANger architecture introduced by Lin et al. [34] represents a current benchmark in time series generation. It tackles a similar problem with TimeGAN as both separate the generation of static attributes from the time series measurements, although focusing on privacy over accurate reproduction of the target distribution time series. Specifically, the generation procedure for the time series implies a conditional process akin to prior work with Conditional Generative Adversarial Nets [35].

The network further improves by providing a normalization approach that avoids mode collapse on long time series: instead of normalizing the entire data set at once, using the global minimum and maximum, the algorithm normalizes using the per-sample minimum and maximum. Furthermore, the minimum and maximum are attached as static metadata describing the associated time series. Thus, the generator is responsible for creating the static features and is also in charge of creating the features that normalize the time series.

The final DoppelGANger architecture uses three networks for generating data: a generator network used for generating static attributes and a generator network used for generating the minimum and maximum of each time series, as described above. The data generated by the two networks is fed into the third, a recurrent neural network which leverages the provided information plus its internal state to generate measurements. A stacked discriminator critiques the generator's work: one model focuses on the generated static features (also called metadata), while the other is a recurrent neural network critiquing the generated measurements.

3. Methodology

This section presents the steps undertaken in obtaining the augmented dataset. The dataset is discussed, along with the data preprocessing operations carried out. Afterwards, the *LSTM-based autoencoder architecture* employed for translating between the initial and the latent space is detailed upon. The latent space obtained is explored, and underrepresented sequences are identified using an *OPTICS clustering algorithm*. Moving on, the *adversarial network* (WGAN) used to sample new examples is presented, and the process of integrating the new samples into the initial dataset is discussed.

A high-level overview of the DAuGAN approach is depicted in Figure 1. The code, models, and dataset used are publicly available in [36].

3.1. Dataset. For the purpose of the paper, a dataset describing the evolution of the price for Apple's company stock, denoted by the AAPL ticker on New York stock exchange, has been chosen.

The dataset presents samples at every 15 minutes, covering the company's price evolution starting from 1st of January 1998 until 3rd of December 2021, totalling 289487 time steps. The dataset features 7 initial columns: date, time, open, high, low, close, and volume. These features are often used in the domain of algorithmic trading and offer indicators on the price's evolution per time step. The open feature describes the price at the start of the time step, and high and low describe the maximum and minimum reached throughout the time-step, while the close price describes the price at interval's end. The volume feature represents the number of trades executed in the given period.

Table 1 presents a sample fragment from the beginning of the time series.

Since the time steps are continuous, there are constraints that apply for any time step t .

$$\forall t \geq 1: \text{close}_{t-1} = \text{open}_t, \quad (1)$$

$$x_t \leq \text{high}_t, \forall t, \forall x \in \{\text{low}, \text{close}, \text{open}\}, \quad (2)$$

$$x_t \geq \text{close}_t, \forall t, \forall x \in \{\text{low}, \text{high}, \text{open}\}. \quad (3)$$

However, real world imposes exceptions to these constraints. First, the data used come only from the trading hours action, starting from 09:30 until 16:00, Monday to Friday, when all traders can take part in the market. However, the NYSE, and in general, the exchanges located in the United States, also present a "before-market" and "after-market" period, limited to institutional investors such as pension funds, hedge funds, or banks. Furthermore, shares can be traded between any two interested parties, without the exchange as an intermediary.

It is beyond the scope of the paper to identify and enumerate all possible sources of discontinuity that could violate Equation (1). The simplifying assumption that the condition holds for any two consecutive steps is made.

Basic statistics for the numerical features of the dataset are calculated. The analysis from Table 2 reveals that volume features a very wide, $[10^2, 10^7]$, domain.

Columns open and volume are plotted, observing that columns high, low, and close will trend in correlation and close to open column, leading to Figure 2. Figure 2(a) presents the histogram of volume column in the initial dataset, while Figure 2(b) depicts the open column evolution in time.

The keen observer will be very interested in the two sudden drops in price illustrated in open column evolution illustrated in Figure 2(b). They represent a domain-specific event called *share splitting*. In a $X: 1$ share split, the price of one share is divided by X while each share presplit is

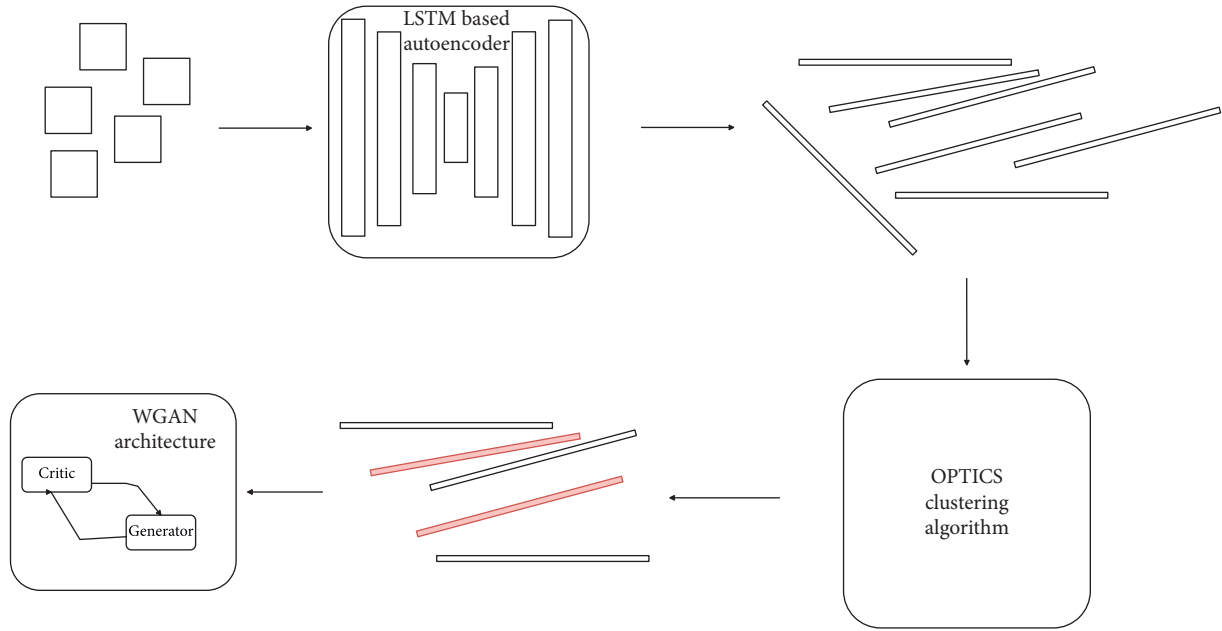


FIGURE 1: DAuGAN approach.

TABLE 1: An excerpt from the beginning of the time series.

Date	Time	Open	High	Low	Close	Volume
1998/02/01	09:30	13.6250	13.7500	13.5000	13.6875	20270
1998/02/01	09:45	13.6875	13.7500	13.5000	13.6250	334000
1998/02/01	10:00	13.6250	13.7500	13.5625	13.7500	299900
1998/02/01	10:15	13.7500	14.0000	13.6250	14.0000	430201
1998/02/01	10:30	13.9375	14.8125	13.7500	14.6250	944200
1998/02/01	10:45	14.6250	14.7500	14.3750	14.4375	218103

TABLE 2: Count, mean, standard deviation, and minimum values for the entire dataset plus upper bounds for each quarter of the dataset in sorted order.

	Open	High	Low	Close	Volume
Count	$2.894870e + 05$	$2.894870e + 05$	$2.894870e + 05$	$2.894870e + 05$	$2.894870e + 05$
Mean	187.757776	188.038462	187.466230	187.758491	$4.580510e + 05$
Std	160.514466	160.689723	160.326415	160.513598	$9.206810e + 05$
Min	12.550000	12.950000	11.312500	12.850000	$1.000000e + 02$
25%	78.750000	78.920000	78.500000	78.750000	$5.900000e + 03$
50%	131.040000	131.330000	130.790000	131.040000	$1.033630e + 05$
75%	248.160000	248.605000	247.625000	248.150000	$5.635505e + 05$
Max	704.800000	705.070000	704.530000	704.800000	$7.514145e + 07$

replaced with X times more. This preserves the value of the investment while lowering the financial bar for buying one share, attracting interest and activity from smaller investors with the better price per action.

3.1.1. Data Preprocessing. Date and time features are discarded, since the augmented dataset obtained at the end of the procedure contains a larger number of samples and the two features must be mocked. For training open, high, low, close, and volume columns are used. Considering the exponential distribution of the volume highlighted in Figure 2, with values

in the domain of $[10^2, 10^7]$, a logarithm transformation is applied column-wise, followed two independent statistical normalization operations: one for the [open, high, low, close] columns and one for volume. Applying normalization on all columns would violate the constraint of Equation (1), as the features follow different distributions. Training on non-logarithmized leads to an autoencoder collapse, with most values outputted by decoder for volume being zero.

Figure 3 illustrates the dataset after normalization. One observes that the transformation of volume feature (Figure 3(a)) is notable, compared to the initial data (Figure 2(a)).

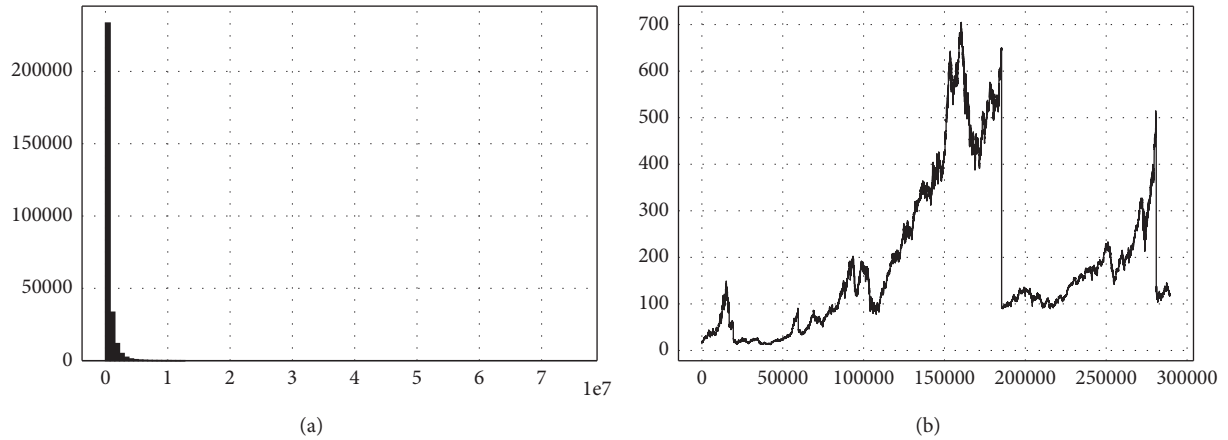


FIGURE 2: (a) Histogram of volume feature in the initial dataset; (b) open feature evolution in time.

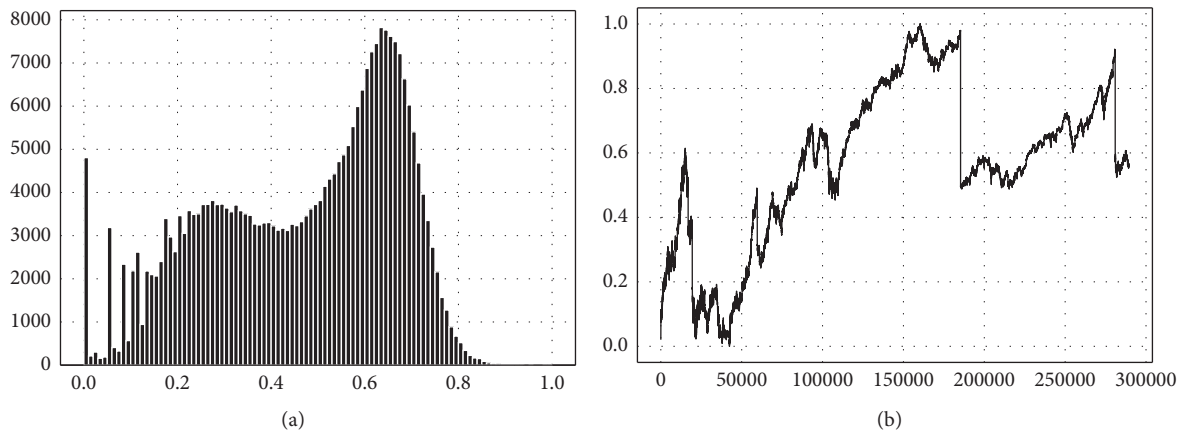


FIGURE 3: (a) Histogram of volume feature in the normalized dataset; (b) open feature evolution in time in the dataset after normalization.

In preparation for the training of the models, time series are split into chunks of twenty time steps and an overlap of eight time steps between two chunks formed of the forty consecutive time steps.

3.2. Autoencoder. It is difficult to identify outliers in the original, time series problem space. The issue is resolved via an autoencoder architecture that translates the time series samples into a latent, Euclidean space, where clustering can be applied in order to identify outliers. The autoencoder is tasked with translating between the two spaces. The dimensions of the latent space are not significant in themselves, and the cardinality has been chosen in order to minimise Pearson correlation, reducing autoencoder training time.

The architecture of the proposed autoencoder is illustrated in Table 3 (the architecture for the encoder) and Table 4 (the architecture for the decoder). The encoder and decoder parts of the architecture mirror each other. When reproducing the experiments, one should expect training and validation losses in the domain of 10^{-4} after 1000 epochs of training. Besides the val_{loss} metric, Person correlation between dimensions of latent space is employed in order to determine the minimum number of nonredundant dimensions.

In the encoder, one-dimensional convolutions use same padding mode, increasing the number of dimensions fed into the LSTM layers. It has been observed that the expansion in dimensionality aids the convergence of LSTM layers. Three stacked LSTM layers are used to capture temporal dependencies and encode them in the condensed latent form. The decoder operates in the same manner, with stacked LSTM layers expanding the temporal correlations encoded in latent and reverse convolutions condensing dimensions back to the original form. Following best practices from the literature, dropout layers with a rate $\beta = 0.3$ are intertwined for regularization purposes [37], and PReLU function is employed as activation between all layers [38]. The optimization algorithm involved in training is Stochastic Gradient Descent with a learning rate of $\alpha = 0.0001$ that uses Nesterov momentum [39]. The choice of the optimizer is motivated by the fact that Adam is not guaranteed to converge [40].

3.2.1. Analysing the Latent Space. The Pearson correlation heat map between the features is illustrated in Figure 4. The latent features have weak correlation, indicating an optimum number of features. An OPTICS algorithm [41] is applied for identifying minority clusters, sampling the epsilon

TABLE 3: Encoder architecture.

Layer (type)	Output shape	Param #
input_1 (InputLayer)	[(None; 20; 5)]	0
conv1d (Conv1D)	(None; 20; 256)	5376
p_re_lu (PReLU)	(None; 20; 256)	5120
dropout (Dropout)	(None; 20; 256)	0
conv1d_1 (Conv1D)	(None; 20; 256)	262400
p_re_lu_1 (PReLU)	(None; 20; 256)	5120
dropout_1 (Dropout)	(None; 20; 256)	0
conv1d_2 (Conv1D)	(None; 20; 256)	262400
p_re_lu_2 (PReLU)	(None; 20; 256)	5120
dropout_2 (Dropout)	(None; 20; 256)	0
lstm (LSTM)	(None; 20; 256)	525312
p_re_lu_3 (PReLU)	(None; 20; 256)	5120
dropout_3 (Dropout)	(None; 20; 256)	0
lstm_1 (LSTM)	(None; 20; 256)	525312
p_re_lu_4 (PReLU)	(None; 20; 256)	5120
dropout_4 (Dropout)	(None; 20; 256)	0
lstm_2 (LSTM)	(None; 12)	12912

TABLE 4: Decoder architecture.

Layer (type)	Output shape	Param #
input_2 (InputLayer)	[(None; 12)]	0
repeat_vector (RepeatVector)	(None; 20; 12)	0
lstm_3 (LSTM)	(None; 20; 256)	275456
p_re_lu_5 (PReLU)	(None; 20; 256)	5120
dropout_5 (Dropout)	(None; 20; 256)	0
lstm_4 (LSTM)	(None; 20; 256)	525312
p_re_lu_6 (PReLU)	(None; 20; 256)	5120
dropout_6 (Dropout)	(None; 20; 256)	0
conv1d_3 (Conv1D)	(None; 20; 256)	262400
p_re_lu_7 (PReLU)	(None; 20; 256)	5120
dropout_7 (Dropout)	(None; 20; 256)	0
conv1d_4 (Conv1D)	(None; 20; 256)	262400
p_re_lu_8 (PReLU)	(None; 20; 256)	5120
dropout_8 (Dropout)	(None; 20; 256)	0
conv1d_5 (Conv1D)	(None; 20; 256)	262400
p_re_lu_9 (PReLU)	(None; 20; 256)	5120
dropout_9 (Dropout)	(None; 20; 256)	0
time_distributed (TimeDistributed)	(None; 20; 5)	1285
time_distributed_1 (TimeDistributed)	(None; 20; 5)	0

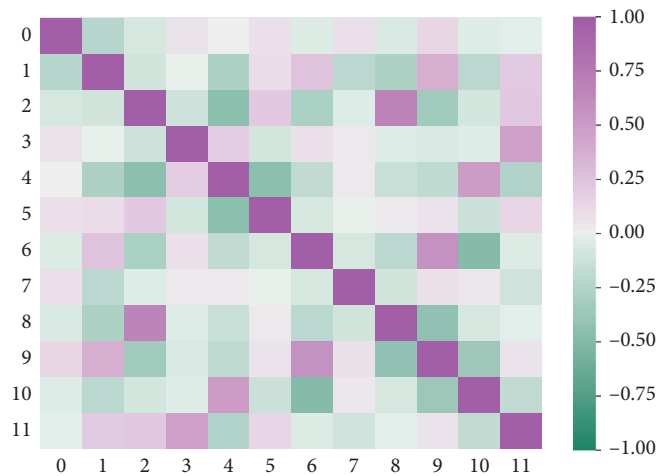


FIGURE 4: Pearson correlation between features.

hyperparameter linearly from the $[0.05, 0.5]$ range. The Elbow method [42] is applied on total variance in order to choose an optimum number of clusters. Analysis yields a majority cluster of ≈ 20000 points, with the rest being classified as outliers or noise. Principal component analysis [43] is employed for reducing dimensionality to three axes, thus allowing visualization.

3.3. Synthesising New Samples. The generative architecture presented in Tables 5 and 6 is able to synthesise credible examples that resemble the minority class. For statistical confirmation, a multivariate Wilcoxon test between the real minority points and the generated points is used [44, 45]. A p -value of 0.7275 is obtained at a significance level $\alpha = 0.05$, unable to refute the null hypothesis, proving there is no significant difference between the real and the generated points. The resemblance between distributions can be observed in Figure 5.

Figure 6 depicts the correlation heat map for the synthesised examples.

It is worth noting that the feature correlation for synthesised latent vectors slightly differs from the minority features' correlation, as shown in Figure 7.

3.3.1. Smoothing the Synthesised Examples. Despite statistical resemblance of the generated samples, the WGAN is unable to model the constraints from formulae (1), (2), and (3).

The issue is approached as an optimization problem: the closest point to the initially synthesised data point is found that satisfies all constraints simultaneously. Bayesian search [46] in tandem with a greedy algorithm is employed. The time steps corresponding to the latent point are iteratively

“smoothed.” If the time step’s values do not respect the imposed constraints, the latent point will be assigned a negative value and a new neighbouring latent point will be verified. Should the time step be found adequate, it is assigned the inverse of the distance between the original location of the sampled synthetic point and the current location of the data point, and this metric is maximised. It should be noted that the original data point is kept as reference throughout all explorations.

Formally, let $d = \|\text{clv} - \text{olv}\|^2$ (clv denotes the current latent vector, and olv represents the original latent vector); Bayesian search must optimize the black-box function γ for each time step, minimising the distance d at the overall data point level. After fixing a time step, the open price of the next time step is set to be equal to the close price of the current one in order to preserve continuity.

$$\gamma(\text{open}, \text{high}, \text{low}, \text{close}) = \begin{cases} -1, & \text{if (time step passes),} \\ \frac{1}{d}, & \text{otherwise.} \end{cases} \quad (4)$$

The search space of the Bayesian process over γ is constrained to $X \pm \sqrt{X}$, where X is the value for any of the features. The point with the maximum score after a set number of steps is selected and integrated in the time series. For time steps with index $t \geq 1$, the search is carried out only for $\{\text{high}, \text{low}, \text{close}\}$, since open price has been set at the first step.

In order to integrate a smoothed chunk ζ into the original time series, a tuple of consecutive chunks, defined by index $t < \text{len}(\text{time_series}) - 1$, is identified such that the objective is minimised:

$$\left\| (\text{close}_t, \text{open}_{t+1}, \gamma(\text{volume}_t: \text{volume}_{t+1})) - (\text{open}_\zeta, \text{close}_\zeta, \gamma(\text{volume}_\zeta)) \right\|^2. \quad (5)$$

Minimising the objective is equivalent to minimising the distance between $(\text{close}_t, \text{open}_\zeta)$ and $(\text{close}_\zeta, \text{open}_{t+1})$, preserving the overall smoothness of the time series. Table 7 presents a chunk fragment obtained in the generation process.

4. Results and Discussion

With the goal of answering research question RQ2, Section 4.1 presents the benchmark used to assess the effectiveness of the augmentation introduced in Section 3 in improving the performance of trading algorithms. The obtained results are then presented in Section 4.2.

4.1. Benchmark Methodology and Data Preparation. In order to assess the impact of augmentation, independent reinforcement learning trading algorithms and domain-specific data preparation procedures are provided by the open-source *FinRL* library [47].

For the benchmark, 80% of the original time series is kept for training the trading strategies, while employing the rest of 20% for blind validation. For fair comparison, the augmentation procedure is employed only on the training portion, and the impact is measured on the validation period. The algorithms are trained on two distinct datasets, the augmented training dataset and original training dataset, until convergence, and score on validation is measured.

FinRL preprocessing is applied on both datasets. The procedure adds technical indicators, whose purpose is to highlight trends in a security’s price evolution, such as relative volatility, magnitude of price shifts, or trends in price shift. The indicators added by *FinRL* are 12-MACD, Bollinger Bands, 30-RelativeStrengthIndex, 30-CommodityChannelIndex, 30-AverageDirectionalIndex, 30-CloseSimpleMovingAverage, and 60-CloseSimpleMovAvg.

The *FinRL* library requires the presence of two extra columns in order to calculate the technical indicators: the tic column which represents the security’s descriptor (*FinRL* is

TABLE 5: WGAN generator architecture.

Layer (type)	Output shape	Param #
input_4 (InputLayer)	[(None; 15; 1)]	0
dense_3 (Dense)	(None; 15; 20)	40
conv1d_9 (Conv1D)	(None; 8; 16)	1296
leaky_re_lu_3 (LeakyReLU)	(None; 8; 16)	0
conv1d_10 (Conv1D)	(None; 4; 16)	1040
leaky_re_lu_4 (LeakyReLU)	(None; 4; 16)	0
flatten_1 (Flatten)	(None; 64)	0
dense_4 (Dense)	(None; 100)	6500
dense_5 (Dense)	(None; 100)	10100
dense_6 (dense)	(None; 12)	1212

TABLE 6: WGAN critic architecture.

Layer (type)	Output shape	Param #
input_3 (InputLayer)	[(None; 12; 1)]	0
conv1d_6 (Conv1D)	(None; 6; 16)	80
leaky_re_lu (LeakyReLU)	(None; 6; 16)	0
conv1d_7 (Conv1D)	(None; 3; 16)	1040
leaky_re_lu_1 (LeakyReLU)	(None; 3; 16)	0
conv1d_8 (Conv1D)	(None; 2; 16)	1040
leaky_re_lu_2 (LeakyReLU)	(None; 2; 16)	0
flatten (Flatten)	(None; 32)	0
dense_1 (Dense)	(None; 100)	3300
dense_2 (Dense)	(None; 1)	101

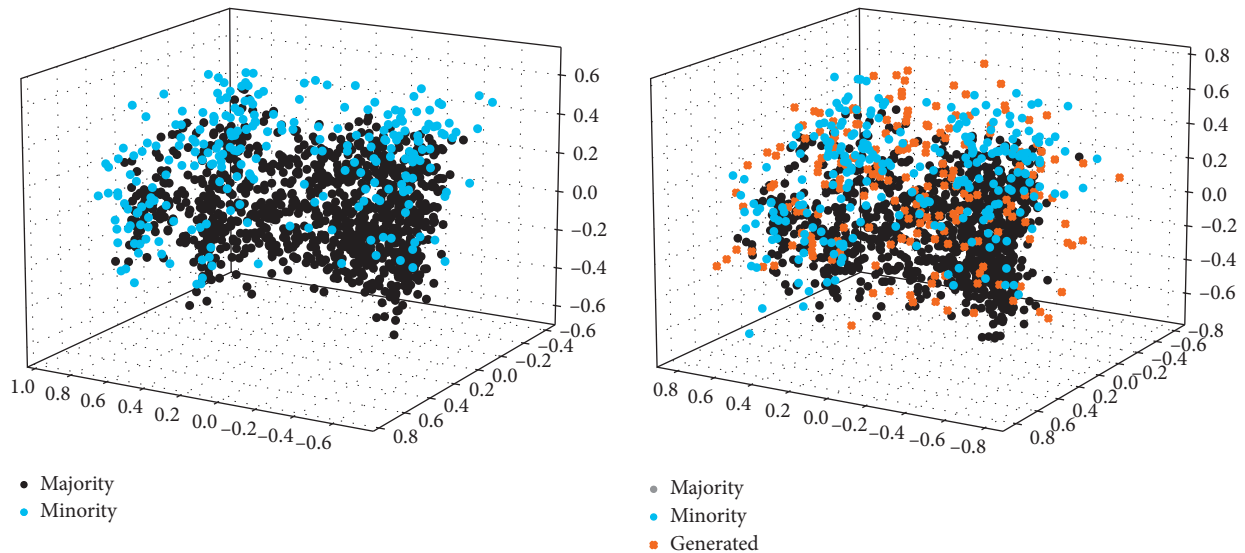


FIGURE 5: Distribution of latent points before and after synthesising new examples.

able to trade multiple securities at once, hence the requirement for this column) and the date column which represents the date and time of the column. With the simplifying assumption that the security has been traded continuously in intervals of 15 minutes starting from an arbitrary date, the two columns are added.

The trading agent's state is described by the tuple (shares, capital), where shares describes the number of owned shares, while capital describes the amount of monetary units

available for buying shares. The state of the agent is completed by the time series, as seen until moment t of time. The initial state $(\text{shares}_0, \text{capital}_0) = (0, 200000)$. The value of a portfolio value with shares is defined as $\text{capital}_t + \sum_{0,r} \text{open}_t$.

At any time step, the agent's action space spans the integers $[-\min(k, \text{shares}_t), k]$. k is a positive integer hyperparameter set in the environment. Negative integers indicate selling that amount of shares, receiving an amount of capital equal to the opening price for each sold share.

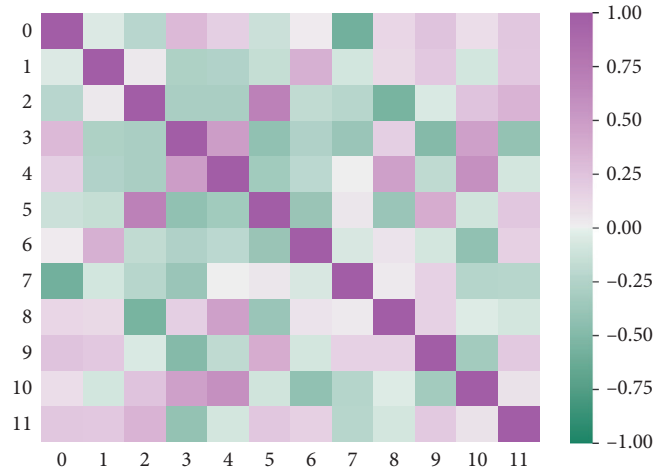


FIGURE 6: Correlation heat map for generated examples.

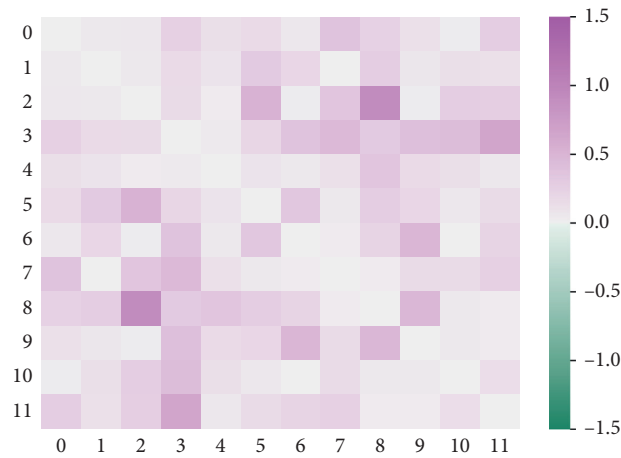


FIGURE 7: Absolute difference between the correlation maps of minority and generated examples.

TABLE 7: Generated chunk fragment.

	Open	High	Low	Close	Volume
0	326.366272	341.629822	314.827942	330.429718	517.638489
1	330.429718	336.826141	310.153687	325.654053	899.540161
2	325.654053	338.448029	311.383545	327.100647	2897.673340
3	327.100647	351.454468	313.967712	351.090942	3598.678955
4	351.090942	340.434845	313.185944	329.017059	4032.858887
5	329.017059	340.508057	313.215607	329.068115	5119.424316
6	329.068115	335.061981	308.940491	324.127441	153.940964
7	324.127441	331.650665	305.705658	320.777008	144.449570
8	320.777008	335.563629	308.537842	324.214722	4437.651367
9	324.214722	340.812073	313.569183	329.397736	3368.686523
10	329.397736	340.041931	312.984558	328.713165	2112.148193
11	328.713165	344.093750	312.746857	319.489136	575.971191
12	319.489136	339.126801	312.318512	327.886292	1199.597290
13	327.886292	344.057251	316.442230	332.473938	9648.517578
14	332.473938	346.419373	318.820251	334.856781	5601.142578
15	334.856781	345.538300	318.073364	334.029663	4347.550781

TABLE 8: Improvement in trading performance, measured with respect to the number of introduced samples and algorithm used.

Algorithm	# of introduced samples		
	2000	4000	6000
DDPG	+2.71%	+3.44%	+4.25%
PPO	+3.12%	+3.47%	+3.82%
A2C	+1.89%	+2.31%	+2.42%

TABLE 9: Hyperparameters for DDPG, PPO, and A2C algorithms.

Algorithm	Hyperparameter	Value
DDPG	critic_learning_rate	$1e-3$
	actor_learning_rate	$1e-3$
	l2_weight_regularization	$1e-6$
	gradient_clipping	None
	train_batch_size	256
	episodes_before_learn_start	1500
PPO	sgd_lr	$5e-5$
	epochs_per_train_batch	30
	ppo_parameter_clipping	0.3
	kl_divergence_target	0.01
	value_function_clip_parameter	10
	entropy_coeff	0.0
A2C	Adam_learning_rate	$1e-4$
	Adam_minibatch_size	32
	batch_size	12500
	gradient_steps	3000
	tau_mov_avg	0.01
	l2_weight_regularization	$1e-6$
	critic_learning_rate	$1e-4$
	actor_learning_rate	$1e-4$

Positive amounts indicate buying stock units and have the reverse effect on capital. The special case $k = 0$, denoting the agent’s choice to hold its current position, should be noted.

4.2. Results. A positive correlation between the amount of samples introduced in the original time series, and the performance improvement of the algorithm is identified. The improvement is defined as the difference in portfolio value between training on the original time series and augmented series.

Table 8 summarizes the performance improvement of DAuGANa. Permutations of introduced samples and benchmark algorithm used: Deep Deterministic Policy Gradient (DDPG), Proximal Policy Optimization (PPO), and Advantage Actor Critic (A2C). Regarding the hyperparameter setting, the algorithms were trained using the values depicted in Table 9.

5. Conclusions and Future Work

The paper has introduced a novel augmentation method for identifying poorly represented sections of a time series, studied the synthesis of new data points and their integration into the time series, and assessed the performance improvement on a benchmark machine learning model.

Data synthetisation is a valid training augmentation in the area of algorithmic trading, which has the potential to be

extended to other domains involving time series, due to the generality of the latent space approach. Of interest for the future are mission-critical tasks such as detecting rare medical conditions or weather now-casting, where performance improvement is vital.

As possible improvements, the authors hypothesise that the use of recurrent neural networks at the generation step [10], combined with the autonormalization trick using DoppelGANger discussed in [34], can result in longer synthesised time series, eliminating the need for interleaving generated samples back into the original time series. Instead, numerous independent “training episodes,” several chunks in length, could be fed to the reinforcement learning agent, a method known to improve training performance [48].

Data Availability

The data used to support the findings of this study are available at the aforementioned repository [36].

Conflicts of Interest

The authors declare no conflicts of interest.

Acknowledgments

The first author acknowledges the financial support received from Babeş-Bolyai University through the special

scholarship for scientific activity for the academic year 2020–2021. The research leading to these results has received funding from the NO Grants 2014–2021, under Project contract no. 26/2020.

References

- [1] C. Shorten and T. M. Khoshgoftaar, “A survey on image data augmentation for deep learning,” *Journal of Big Data*, vol. 6, no. 1, 2019.
- [2] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “Smote: synthetic minority over-sampling technique,” *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, 2011.
- [3] V. Sandfort, K. Yan, P. J. Pickhardt, and R. M. Summers, “Data augmentation using generative adversarial networks (CycleGAN) to improve generalizability in CT segmentation tasks,” *Scientific Reports*, vol. 9, no. 1, 2019.
- [4] M. Mustafa, B. Deborah, B. Wahid, L. Zarija, A.-R. Rami, and M. K. Jan, “CosmoGAN: creating high-fidelity weak lensing convergence maps using Generative Adversarial Networks,” *Comput. Astrophys.* vol. 6, p. 1, 2019.
- [5] T. Karras, S. Laine, and T. Aila, “A style-based generator architecture for generative adversarial networks,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4396–4405, IEEE Computer Society, Los Alamitos, CA, USA, June 2019.
- [6] M. Pasini, “MelGAN-VC: voice conversion and audio style transfer on arbitrarily long samples using spectrograms,” *Electrical Engineering and Systems Science*, <https://arxiv.org/abs/1910.03713>, 2019.
- [7] M. Chauvet, “Stock market fluctuations and the business cycle,” *SSRN Electronic Journal*, 2001.
- [8] S. Edwards, J. G. Biscarri, and F. P. de Gracia, “Stock market cycles, financial liberalization and volatility,” *Technical Report D*, 2003.
- [9] N. Tavakoli, S. Siami-Namini, M. A. Khanghah, F. M. Soltani, and A. S. Namin, “Clustering time series data through autoencoder-based deep learning models,” 2020, <https://arxiv.org/abs/2004.07296>.
- [10] J. Yoon, D. Jarrett, and M. van der Schaar, “Time-series generative adversarial networks,” Edited by H. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché-Buc, E. Fox, and R. Garnett, Eds., in *Proceedings of the Advances in Neural Information Processing Systems*, vol. 32, Curran Associates, Inc., Vancouver, Canada, December 2019.
- [11] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena, “Self-attention generative adversarial networks,” Edited by K. Chaudhuri and R. Salakhutdinov, Eds., in *Proceedings of the 36th International Conference on Machine Learning, Ser. Proceedings of Machine Learning Research*, vol. 97, pp. 7354–7363pp. 7354–, Long Beach, CA, USA, June 2019.
- [12] I. J. Goodfellow, “NIPS 2016 tutorial: generative adversarial networks,” 2017, <http://arxiv.org/abs/1701.00160>.
- [13] G. B. Goh, N. O. Hodas, and A. Vishnu, “Deep learning for computational chemistry,” *Journal of Computational Chemistry*, vol. 38, no. 16, pp. 1291–1307, 2017.
- [14] J. Kolen, *A Field Guide to Dynamical Recurrent Networks*, IEEE Press, New York, NY, USA, 2001.
- [15] N. Kodali, J. D. Abernethy, J. Hays, and Z. Kira, “How to train your DRAGAN,” 2017, <http://arxiv.org/abs/1705.07215>.
- [16] J. Zhu, T. Park, P. Isola, and A. A. Efros, “Unpaired image-to-image translation using cycle-consistent adversarial networks,” 2017, <http://arxiv.org/abs/1703.10593>.
- [17] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, “Unpaired image-to-image translation using cycle-consistent adversarial networks,” in *Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 2242–2251, IEEE, Venice, Italy, October 2017.
- [18] M. Amodio, S. Krishnaswamy, and “Travelgan, “Image-to-image translation by transformation vector learning,” 2019, <http://arxiv.org/abs/1902.09631>.
- [19] M. Amodio and S. Krishnaswamy, “TraVeLGAN: image-to-image translation by transformation vector learning,” in *Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8975–8984, IEEE, Long Beach, CA, USA, June 2019.
- [20] M. Arjovsky, S. Chintala, and L. Bottou, “Wasserstein generative adversarial networks,” in *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, pp. 214–223, Sydney, Australia, August 2017.
- [21] D. Bank, N. Koenigstein, and R. Giryes, “Autoencoders,” 2020, <https://arxiv.org/abs/2003.05991>.
- [22] K. Pearson and L. I. I. I. “On lines and planes of closest fit to systems of points in space,” *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 2, no. 11, pp. 559–572, 1901.
- [23] G. E. Hinton, “Reducing the dimensionality of data with neural networks,” *Science*, vol. 313, no. 5786, pp. 504–507, 2006.
- [24] I. Goodfellow, *Deep Learning*, The MIT Press, Cambridge, MA, USA, 2016.
- [25] R. Salakhutdinov and G. Hinton, “Semantic hashing,” *International Journal of Approximate Reasoning*, vol. 50, no. 7, pp. 969–978, 2009.
- [26] C. Hansen, C. Hansen, J. G. Simonsen, S. Alstrup, and C. Lioma, “Unsupervised neural generative semantic hashing,” <http://arxiv.org/abs/1906.00671>.
- [27] C. Yu, J. Liu, and S. Nemati, “Reinforcement learning in healthcare: a survey,” 2019, <http://arxiv.org/abs/1908.08796>.
- [28] L. Wang, W. Zhang, X. He, and H. Zha, “Supervised reinforcement learning with recurrent neural network for dynamic treatment recommendation,” in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, ser. KDD ’18*, pp. 2447–2456, Association for Computing Machinery, New York, NY, USA, July 2018.
- [29] Y. Li, Y. Wen, D. Tao, and K. Guan, “Transforming cooling optimization for green data center via deep reinforcement learning,” *IEEE Transactions on Cybernetics*, vol. 50, no. 5, pp. 2002–2013, 2020.
- [30] D. Silver, A. Huang, C. J. Maddison et al., “Mastering the game of go with deep neural networks and tree search,” *Nature*, vol. 529, pp. 484–489, 2016.
- [31] H. Yang, X.-Y. Liu, S. Zhong, and A. Walid, “Deep reinforcement learning for automated stock trading: an ensemble strategy,” in *Proceedings of the ICAIF ’20: ACM International Conference on AI in Finance*, Association for Computing Machinery, New York, NY, USA, October 2020.
- [32] T. Théate and D. Ernst, “An application of deep reinforcement learning to algorithmic trading,” *Expert Systems with Applications*, vol. 173, Article ID 114632, 2021.
- [33] D. Mehta, “State-of-the-Art reinforcement learning algorithms,” *International Journal of Engineering Research and Technology*, vol. 8, pp. 717–722, 2020.
- [34] Z. Lin, A. Jain, C. Wang, G. C. Fanti, and V. Sekar, “Generating high-fidelity, synthetic time series datasets with doppelganger,” <http://arxiv.org/abs/1909.13403>.

- [35] M. Mirza and S. Osindero, "Conditional generative adversarial nets," 2014, <http://arxiv.org/abs/1411.1784>.
- [36] A. Bratu, "Daugan paper code and dataset repository," 2021, <https://github.com/andreibratu/bachelor/tree/master/thesis>.
- [37] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, and "Dropout, "A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.
- [38] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: surpassing human-level performance on imagenet classification," <http://arxiv.org/abs/1502.01852>.
- [39] A. Botev, G. Lever, and D. Barber, "Nesterov's accelerated gradient and momentum as approximations to regularised update descent," in *Proceedings of the 2017 International Joint Conference on Neural Networks (IJCNN)*, May 2017.
- [40] S. J. Reddi, S. Kale, and S. Kumar, "On the convergence of Adam and beyond," in *Proceedings of the International Conference on Learning Representations*, Vancouver, Canada, May 2018.
- [41] M. Ankerst, M. M. Breunig, H. Peter Kriegel, and J. Sander, "Optics: ordering points to identify the clustering structure," in *Proceedings of the . ACM SIGMOD'99 Int. Conf. on Management of Data*, pp. 49–60, ACM Press, Philadelphia, PA, USA, June 1999.
- [42] M. A. Syakur, B. K. Khotimah, E. M. S. Rochman, and B. D. Satoto, "Integration k-means clustering method and elbow method for identification of the best customer profile cluster," *IOP Conference Series: Materials Science and Engineering*, vol. 336, Article ID 012017, 2018.
- [43] I. T. Jolliffe and J. Cadima, "Principal component analysis: a review and recent developments," *Philosophical Transactions of the Royal Society A: Mathematical, Physical & Engineering Sciences*, vol. 374, no. 2065, Article ID 20150202, 2016.
- [44] H. Oja and R. H. Randles, "Multivariate nonparametric tests," *Statistical Science*, vol. 19, no. 4, 2004.
- [45] C. fan Sheu and S. O'Curry, "Implementation of nonparametric multivariate statistics with s," *Behavior Research Methods, Instruments, & Computers*, vol. 28, no. 2, pp. 315–318, 1996.
- [46] R. Turner, D. Eriksson, M. McCourt et al., "Bayesian optimization is superior to random search for machine learning hyperparameter tuning: analysis of the black-box optimization challenge 2020," 2021, <https://arxiv.org/abs/2104.10201>.
- [47] X.-Y. Liu, H. Yang, Q. Chen et al., "FinRL: a deep reinforcement learning library for automated stock trading in quantitative finance," *SSRN Electronic Journal*, <https://arxiv.org/abs/2011.09607>, 2020.
- [48] V. Mnih, K. Kavukcuoglu, D. Silver et al., "Playing atari with deep reinforcement learning," 2013, <http://arxiv.org/abs/1312.5602>.

Research Article

A Clustering-based Method for Business Hall Efficiency Analysis

Tianlin Huang  and Ning Wang 

College of Information and Smart Electromechanical Engineering, Xiamen Huaxia University, Xiamen 361024, China

Correspondence should be addressed to Ning Wang; nwang97@163.com

Received 20 May 2021; Revised 11 August 2021; Accepted 7 September 2021; Published 1 October 2021

Academic Editor: Antonio J. Peña

Copyright © 2021 Tianlin Huang and Ning Wang. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Excessive or insufficient business hall resources may result in unreasonable resource allocation, adversely affecting the value of an entity business hall. Therefore, proper characteristic parameters are the key factors for analyzing the business hall, which strongly affect the final analysis results. In this study, a characteristic analysis method for the economic operation of a business hall is developed and the feature engineering is established. Because of its simplicity and versatility, the k -means algorithm has been widely used since it was first proposed around 50 years ago. However, the classical k -means algorithm has poor stability and accuracy. In particular, it is difficult to achieve a suitable balance between of the centroid initialization and the clustering number k . We propose a new initialization (LSH- k -means) algorithm for k -means clustering. This algorithm is mainly based on locality-sensitive hashing (LSH) as an index for computing the initial cluster centroids, and it reduces the range of the clustering number. Furthermore, an empirical study is conducted. According to the load intensity and time change of the business hall, an index system reflecting the optimization analysis of the business hall is established, and the LSH- k -means algorithm is used to analyze the economic operation of the business hall. The results of the empirical study show that the LSH- k -means that the clustering method outperforms the direct prediction method, provides expected analysis results as well as decision optimization recommendations for the business hall, and serves as a basis for the optimal layout of the business hall.

1. Introduction

An entity business hall is where a company directly conducts specific business activities, such as commodity trade, business handling, and service. However, owing to rapid urbanization and economic development, unreasonable resource allocation is becoming increasingly prevalent. For example, the number of entity business halls is excessive in some places and insufficient in others. Hence, the deployment of new commercial outlets (halls) or resource allocation optimization for existing retail outlets often needs to be performed manually. Therefore, how to evaluate the efficiency of business halls has emerged as a major concern for many enterprises.

To this end, many researchers have attempted to overcome the disadvantages of human judgment, which is highly subjective. Brandeau and Chiu [1] considered the transportation cost and the distance between the warehouse and the customer and used a gradient-like algorithm to study the

location issue. Wang et al. [2] used nearest-neighbor clustering and the function of Ripley [3] to analyze the layout of commercial outlets and suggested that business type, land price, and traffic accessibility are the critical factors. Gerard [4] analyzed the service needs and waiting demand of customers for bank halls and attempted to shorten the perceived waiting time of customers on the basis of the customers' business types. Thus, customer satisfaction was improved. Anderson et al. [5] used the queuing model to optimize the queuing service system of banks. They determined the optimal number of service windows by acquiring and presenting a large amount of data. Lin et al. [6] studied the relationship between retail stores and street centrality and pointed out that besides the transport network, which has a strong impact on the retailer's location, the street centrality influences the type of retail store. Kang [7] analyzed the changes in warehouses from central urban areas to the urban periphery over time and studied the main factors affecting the warehouse location. Hui [8] used data mining

to establish the channel analysis model for an electricity business hall and optimized the resource allocation. Based on the statistics of customer queuing time, business processing time, customer satisfaction, and so on, Yan et al. [9] established an intelligent access platform for the business data and improved the service efficiency. However, there is no unified standard for the business hall index system.

Clustering is a key technique in data mining, and its applications include pattern recognition [10, 11], image processing [12], and recommendation [13]. Clustering aims to partition data into different categories based on a measure of similarity. The k -means algorithm is widely used owing to its simplicity and effectiveness. However, the different settings of the parameters and random selection of the initial clustering centers make the classical k -means algorithm unstable.

The classical clustering algorithm involves two problems: the first problem is to classify a given dataset on the basis of the prespecified cluster number k ; hence, the problem of determining the “correct cluster number” has attracted considerable interest. Although several methods have been developed for estimating the number of data clusters [14–17], it is difficult to use them in practical applications. Therefore, determining the correct number of clusters has long been an important research topic in cluster analysis. The second problem is to determine the initial clustering center, which has a significant impact on the clustering effect. Studies conducted thus far have explored several initialization methods for the *thek*-means algorithm. For example, the k -means++ algorithm [18] has been proposed to avoid this issue. This algorithm randomly selects the first centroid, and the other centroids are selected as far away as possible from the first centroid. However, random selection is still widely used in practice [19]. Erisoglu et al. [20] proposed an incremental approach for computing the initial clustering centers. In this approach, the reduced dataset is partitioned until the number of clusters equals the predefined number of clusters. However, the number of clusters must be known in advance. The compressed k -means (CKM) algorithm [21] is initialized by locality-sensitive hashing (LSH) [22], and the distance is calculated using the Hamming distance between binary codes. The LSH link [23] can rapidly find a nearby cluster to be connected through the LSH algorithm. David et al. [24] proposed a new LSH scheme adapted to the x^2 distance for approximate nearest neighbors (ANN) search in high-dimensional spaces.

In summary, there is no unified standard for the index system of business halls at present. Therefore, we establish an index system for analyzing the efficiency of a business hall. To address the problem of k -means initialization sensitivity as well as the difficulty in determining the number of clusters, we initialize the k -means centroid on the basis of LSH. Accordingly, we implement the relevant algorithms and present the optimal allocation scheme for the business hall.

The main contributions of this study are as follows:

- (1) According to the average waiting time, ticketing time, and business type of a business hall, we analyze

the average load rate of the business hall and use the relevant characteristic variables to describe the load of the business hall. Finally, we propose a general business hall index system.

- (2) By combining the characteristics of k -means and LSH, We propose a new initialization (LSH- k -means) algorithm for k -means clustering. The model can get the load classification of each business hall by inputting the relevant index variables for the optimization of business hall distribution.
- (3) The results of our empirical analysis verify the validity of the proposed LSH- k -means approach. Thus, LSH- k -means can be efficiently used for the operational analysis of a business hall.

The remainder of this paper is organized as follows: Section 2 introduces the required preliminaries, definitions, and models. Section 3 describes the proposed initialization methodology. Section 4 presents, compares, and discusses the experimental results. Finally, Section 5 concludes the paper.

2. Preliminaries

2.1. k -means Algorithm. The notations used in this paper are defined in Table 1. The k -means [25] method is the most well-known clustering method because of its simplicity. It has been identified as one of the top 10 algorithms in data mining [26]. Given a dataset $D = \{x_1, x_2, \dots, x_n\}$, k -means aims to partition it into k different clusters $C = \{C_1, C_2, \dots, C_k\}$, where $k \leq n$ is a predefined number. The objective of the k -means clustering algorithm is to minimize the sum of squared errors (SSE) [27] over all k clusters. The SSE is defined as follows:

$$\text{SSE} = \sum_j \sum_{x \in C_j} \|x - \Omega_j\|_2^2, \quad (1)$$

where Ω_j denotes the j -th cluster centroid, which is computed as the mean of points in C_j , and $x \in C_j$ is the data object in the j -th cluster.

$$\Omega_j = \frac{1}{|C_j|} \sum_{x_i \in C_j} x_i, \quad (2)$$

where $|C_j|$ denotes the number of data points in the j -th cluster.

To solve equation (1), an expectation–maximization (EM)-like optimization method is adopted by updating $I(x)$ or Ω and simultaneously fixing the other [28]. In general, the clustering procedure involves three steps: (1) initialize k cluster centroids; (2) assign each sample to its closest centroid; and (3) recompute the cluster centroids with the assignments produced in Step 2 and go back to Step 2 until convergence. This is known as the Lloyd iteration procedure [29]. Such an iterative optimization approach has several drawbacks. First, it is sensitive to the initialization, which may lead to an inferior result for a given poorly initialized Ω . Many methods have been proposed to obtain a stable solution, including the k -means++ algorithm [18]. Second,

TABLE 1: The notations used throughout this paper.

Notation	Representation
D	The training set with N points and d dimensions
x_i	The data point
C_j	The j -th clusters
SSE	The sum of squared errors
Ω_j	Denotes the j -th cluster centroid
$ C_j $	The number of data points in the j -th cluster
$O(tkn d)$	Denotes the complexity of k -means
H	A family of hash functions
Pr	The probability
$\text{sim}(x, y)$	The similarity between x and y
M	The maximum load of business hall
p_i	The proportion of a specific business
w_i	The average time to process the business
n	The number of business types
K	The ratio of actual daily load to maximum load
S_w	The working time
S	The number of staffs in the business hall
A_i	Represents the actual load during peak period of the business hall
MS	The ratio of average load to the maximum load
W	The number of working days
AT	The actual load trend
$f(l)$	Denotes the actual load curve fitting function
b_0	A constant
b_1	The regression coefficient
BH	The proportion of high-value business
HBV_i	The high-value business volume
TBV_i	The total business volume
FH	The high-frequency load
h	The high threshold
l	The low threshold
T_{now}	Represent the current time
T_{latest}	Represent the latest time
FL	The low-frequency load
RH	The latest high-load interval
RL	The latest low-load interval
U	The copy training set
N	The size of training dataset
T	The minimum number data point in one cluster
L	The maximum distance in one cluster
B_n	A two-dimensional array
Q	The nearest neighbor data
$\text{query}(x_i)$	Calculate the similarity or distance between x_i
k	The number of clusters
$d(x_i, x_j)$	Distance between x_i and x_j
KM	Conduct k -means algorithm
d_i	The i -th dataset
R^+	Represents the sum of rank, which better than the other
R^-	Opposite to R^+
$S(U, k)$	The tight and separative indicator
XB	The XB index
Obj_{\min}	Objective function
F	The quantitative value of factors
h_i	The weight

finding the optimal solution to k -means is an NP-hard problem. Some variants of k -means have been proposed, such as various parametric k -means, including fuzzy c -means [30, 31]. Third, k -means cannot handle new data, which requires the entire dataset to be observed. The

complexity is $O(t \cdot k \cdot n \cdot d)$, where t , n , k , and d denote the number of iterations, size of the dataset, number of clusters, and dimensionality, respectively. This complexity is considerably higher than that of other well-known clustering algorithms such as DBSCAN [32] and mean shift [33].

2.2. *LSH*. LSH is a well-known solution for the approximate nearest neighbor problem in high-dimensional spaces. LSH was first introduced for the Hamming metric by Indyk and Motwani [34]. Data points are assigned to individual hash buckets in each hash function. The idea of LSH is that closer data points are mapped to the same hash bucket with high probability. LSH has been shown to be effective even for high-dimensional data, both theoretically and experimentally [35]. H are a family of hash functions. Each hash function H must satisfy the LSH property: $\Pr[H(x) = H(y)] = \text{sim}(x, y)$, where $\text{sim}(x, y)$ is the similarity between x and y . These hash functions must meet the following two conditions:

- (1) If $d(x, y) \leq r$, then $\Pr[H(x) = H(y)] \geq p_1$
- (2) If $d(x, y) \geq cr$, then $\Pr[H(x) = H(y)] \leq p_2$

where $d(x, y)$ represents the distance measure between x and y , $r > 1$, and $c < 1$. The definition implies that x and y are hashed into the same bucket in the projection with a very high probability $\geq p_1$. Regardless of whether they are close to each other, they will be hashed into the same bucket $\leq p_2$ with a low probability. A (r, cr, p_1, p_2) -sensitive family of hash functions is useful when the collision probabilities p_1, p_2 satisfy $p_1 > p_2$. Figure 1 shows an example of hashing key space.

3. Proposed LSH-Based Initialization Algorithm

The proposed framework involves three steps: (1) an index system for the efficiency analysis of a business hall is established in Section 3.1. (2) To overcome the problems of poor stability and low accuracy of the classical algorithm, a boost k -means algorithm based on LSH initialization is proposed in Section 3.2. (3) The k -means algorithm is implemented to obtain the clustering results. The details of these three steps are illustrated in Figure 2.

3.1. *Establishment of the Index System*. Through the load analysis of the business hall, we can determine the high and low loads and optimize the business hall. The average utilization rate of each business hall is analyzed according to multiple indicators (including average waiting time, ticketing time for business, and business type). Thus, we can use the relevant characteristic variables to describe the load of the business hall. By applying the clustering algorithms, we can obtain the load categories of different business halls, which provides a basis for planning the locations of the business halls. First, the following two essential features are extracted: the maximum load of the business hall (M) and the ratio of the actual daily load to the maximum load (K).

- (1) The maximum load of the business hall (M) is given by

$$M = \frac{S \times S_w}{\sum_{i=1}^n p_i w_i}, \quad (3)$$

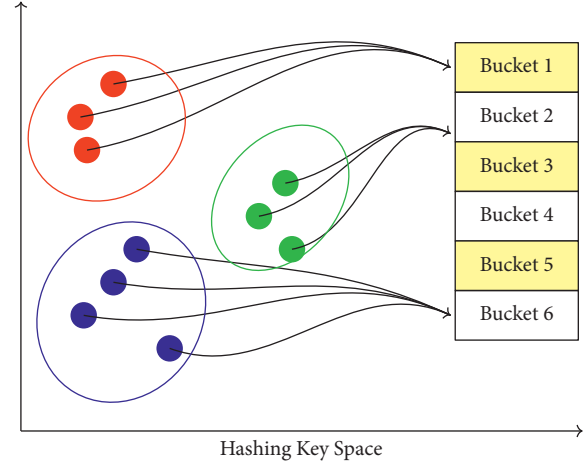


FIGURE 1: Illustration of hashing key space.

where p_i is the proportion of a specific business, w_i is the average time for the clerk to handle the business, n is the number of business types, S_w is the working time of the clerk, and S is the number of clerks in the business hall. The variables are taken from the peak period. This value represents the maximum business volume that a business hall can withstand during the peak period. The peak period can be obtained by measuring the historical data of each business hall.

- (2) The ratio of the actual daily load to the maximum load (K) is given by

$$K = \frac{A}{M}, \quad (4)$$

where A represents the actual daily load of the business hall and M is the maximum load in one day.

By combining the essential characteristics of the business hall and based on the analysis of historical data, we can obtain the calculation indicators of the business hall to prepare for the subsequent model input. Therefore, the feature engineering for the business hall efficiency analysis is established, and the critical indexes extracted are as follows:

- (1) The ratio of the average load to the maximum load (MS) is given by

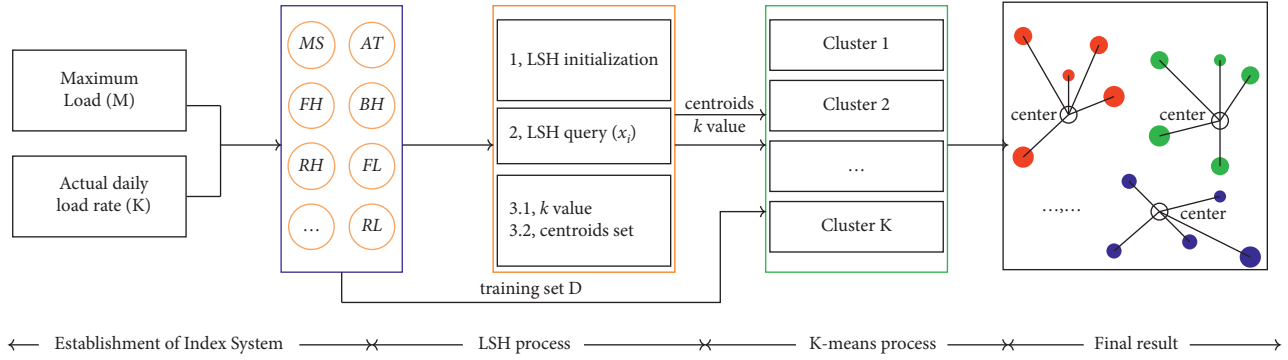
$$MS = \frac{\sum_{i=1}^W A_i}{(W \times M)}, \quad (5)$$

where W is the number of days, and A and M are the same as above. This index denotes the ratio of the average actual load to the maximum load over some time.

- (2) The actual load trend (AT) is given by

$$f(l) = b_0 + b_1 t, \quad (6)$$

$$AT = b_1 + \frac{\sum f(l) \times t - n \times f(l) \times t}{\sum t_i^2 - n \times t^2},$$


 FIGURE 2: Architecture of the proposed LSH k -means model.

where $f(l)$ denotes the actual load curve fitting, b_0 is a constant, b_1 is the regression coefficient, t is a time-independent variable, and n is the number of statistical data. This index indicates whether the load trend of the business hall will be rising, flat, or declining for some time. A fitting curve $f(l)$ can be used to characterize the load trend of the business hall over some time, and the slope b_1 represents the trend state. Our method includes a commercial center, residential center, new urban area, and other factors.

- (3) The proportion of high-value business (**BH**) is given by

$$\text{BH} = \frac{\sum_{i=1}^W \text{HBV}_i}{\sum_{i=1}^W \text{TBV}_i}, \quad (7)$$

where HBV_i is the high-value business volume and TBV_i is the total business volume. Thus, this index denotes the proportion of high-value business to total business in the peak period.

- (4) The high-frequency load (**FH**) is given by

$$\text{FH} = \frac{\text{count}(K_i > h)}{W}, \quad i = 1, 2, \dots, W, \quad (8)$$

where h is a high threshold and FH represents the load of K exceeding h within a period. Furthermore, h can be obtained by statistical analysis of the historical data of the business hall.

- (5) The low-frequency load (**FL**) is given by

$$\text{FL} = \frac{\text{count}(K_i < l)}{W}, \quad i = 1, 2, \dots, W, \quad (9)$$

where l is a low threshold and FL denotes the frequency that K is less than l for some time. Furthermore, l can be obtained by statistical analysis the of historical data of the business hall.

- (6) The latest high-load interval (**RH**) is given by

$$\text{RH} = T_{\text{now}} - T_{\text{latest}(K_i > h)}, \quad i = 1, 2, \dots, W, \quad (10)$$

where T_{now} represents the current time, h denotes a high threshold, and $T_{\text{latest}(K_i > h)}$ refers to the time

when the latest K is greater than h . Furthermore, RH denotes the interval from T_{now} to $T_{\text{latest}(K_i > h)}$.

- (7) The latest low-load interval (**RL**) is given by

$$\text{RL} = T_{\text{now}} - T_{\text{latest}(K_i < l)}, \quad i = 1, 2, \dots, W, \quad (11)$$

where T_{now} represents the current time, l denotes a low threshold, and $T_{\text{latest}(K_i > h)}$ refers to the time when the latest K is greater than h . Furthermore, RL denotes the interval from T_{latest} to $T_{\text{latest}(K_i > h)}$.

3.2. LSH- k -Means. The main purpose of clustering is to divide data into clusters in which objects in the same cluster are close to one another, whereas objects in different clusters are far from one another. Two factors affect the quality of k -means clustering. Before applying the algorithm, we need to specify the number of clusters k and select the initial cluster centroid. Selecting an appropriate initial cluster centroid can improve the quality of clustering. To this end, a critical study was conducted by Vassilvitskii et al. [18, 36]. If the initial cluster centroid is selected carefully, the k -means algorithm converges to a better local optimal solution. Furthermore, careful selection of the initial cluster centroid makes the k -means iteration converge faster [18]. However, to make the initial centroid adapt to the data distribution, it is necessary to scan k rounds. Therefore, although the number of scanning wheels in [36] has been reduced to a small value, the additional computing cost is still inevitable. Our algorithm exploits LSH. The algorithm minimizes the path by adding the nearest neighbor, and LSH can effectively search for the nearest group features in the path. The average time complexity of the hash-based search is $O(1)$. LSH scans the data records and finds the nearest points; the average values are computed after the nearest points are classified as a category. Algorithm 1 describes the process of obtaining the initialization centroids in our proposed LSH- k -means scheme. The main steps are as follows:

- (1) Suppose that we have a set of points $D \in \mathbb{R}^d$ via the index system in Section 3.1. We use LSH to index the feature vectors extracted from the dataset D to reduce the search time for the nearest neighbor of each query. This is based on the hash mapping function, hash functions, and hash table L [37]. Constructing

an effective LSH index structure for approximate nearest neighbor search depends on the number of hash tables L and the number of bits V of the hash codes.

- (2) To facilitate the statistics of nonclustered data points, in Algorithm 1, we copy a dataset U from D . Randomly select one data point x_i from U as the centroid. Then, x_i is merged into the set A_n and removed from the dataset U , where n is the n -th cluster. After obtaining x_i points, query the corresponding bucket number according to the hash table L in Step 1 and take out the data in bucket number V . Calculate the similarity or distance between x_i and the data points x_j in the bucket and return the nearest neighbor data $Q = \text{query}(x_i)$.
- (3) Take data point x_j from Q , whose distance to x_i does not exceed L . Put x_j merged into B_n , that is, $B_n = B_n \cup x_j$, and remove it from the dataset U .
- (4) Repeat Step 3 until the other data point x_j in Q reaches a certain threshold; the threshold can be computed as follows:

$$L \geq d(x_i, x_j) \quad (12)$$

- (5) Repeat Steps 2-3 until the length of the U dataset is less than the threshold L . As shown in Algorithm 1, $\text{count}(U) \leq T$.
- (6) The arithmetic mean values for the final k sets of samples are computed; then, we can obtain the clustering centers for all the categories in this way:

$$C_n = \frac{\text{Sum}(B_n)}{|B_n|}, \quad n = 1, 2, \dots, k. \quad (13)$$

Therefore, based on the aforementioned steps, we will have two algorithms to choose from: “best” movement and “fast” movement [38]. For the “best” movement, we can use equation (13) and the k value in Algorithm 1 as the initial clustering center of the classical k -means input $\mathbf{KM}(k, c_i)$, and run the algorithm; the result is the final result. For “fast” movement, the divided categories can be regarded as approximate clustering results and directly used as the classification results. Because the initial clustering center is determined and the initial category is obtained, the result of the algorithm is more stable and accurate, and it requires a relatively short running time.

4. Experimental Results

First, we use the UCI <https://archive-beta.ics.uci.edu/ml/datasets/datasets> [39] to verify the performance of the proposed algorithm, and we state the verification criteria.

In addition, we use the Mall-Customers dataset <https://www.kaggle.com/shwetabh123/mall-customers> for the value range of the number of clusters k of the proposed LSH- k -means model. Our experimental results demonstrate the effectiveness and superiority of the proposed LSH- k -means. Then, we compare it with the actual business hall dataset and present an example to optimize the business hall operation.

4.1. Experimental Design. To verify the aforementioned points and evaluate the effectiveness of the proposed LSH- k -means model, numerous experiments were conducted on the UCI datasets, which consist of Balance, Wine, Breast, Diabet, Iris, Hayes-roth, Tic-tac-toe, and Bupa. We followed the experiments conducted in a previous study [40]. We briefly review the existing baselines as follows:

- (1) k -means [25] is derived from the classical k -means.
- (2) Enhanced k -means [38] enhances the classical k -means algorithm. The initial cluster centers are determined in advance instead of random selection.
- (3) The AC algorithm [41] for clustering can assume each sample as a pattern; by computing the similarity between patterns, the more similar patterns are grouped into one class, and the less similar patterns are classified into different classes. The difference between two patterns in AC clustering is usually measured by the distance function, including the Euclidean distance or Hamming distance. In the experiment, the AC algorithm is implemented by the KnowledgeMiner Software [41].

There are 200 samples in the Mall-Customers dataset. It includes gender, customer ID, age, annual income, and expenditure scores. In addition, it collects insights from the data and groups them according to their behaviors. The elbow method [42] is a well-known method for determining the optimal value of k . As shown in Figure 3(a), the optimum number of clusters of the Mall-Customers dataset is 5. According to Algorithm 1, we set the minimum number $T = 20$ and the maximum distance $L = 50.0$. Owing to the small amount of data, we set the number of buckets to 1. After 10 LSH-based initializations, we get the value of k between [4–6]. Figure 3(b) shows the results of LSH k -means clustering. The black dots represent the centroids.

There are 525 samples in the Balance dataset. For the classical k -means algorithm, the number of clustering categories that match the real categories is 271, and the matching rate is 51.62%. The corresponding values of the LSH k -means algorithm are 288 and 54.87%, respectively. Similarly, the results of the other UCI datasets are listed in Table 2. To determine whether there are significant differences between algorithms, we use the Wilcoxon signed-rank test [43]. It is a nonparametric statistical test. The Wilcoxon

Required: Training dataset $D \in \mathbb{R}^d$; the size of dataset N ; the minimum number T of data points in one cluster; the maximum distance in one cluster L ; the closest set B_n , where $n \in \{1, 2, \dots, k\}$ and A is a two-dimensional array; k is the number of clusters.

Output: The final clustering result.

- (1) Initialize LSH.
- (2) Index dataset D via LSH.
- (3) Let $0 \leftarrow k$.
- (4) Let $U = D // \text{copy } D$
- (5) **while** $\text{count}(U) \leq T$ **do**
- (6) $k \leftarrow k + 1$.
- (7) Randomly select one point x_i from dataset U .
- (8) $B_n = B_n \cup x_i$.
- (9) x_i is removed from U .
- (10) $Q = \text{query}(x_i)$.
- (11) Let $0 \leftarrow q$
- (12) **for** $q \leq \text{count}(Q)$ **do**
- (13) $q \leftarrow q + 1$.
- (14) **if** $L \geq d(Q_q)$ **then**
- (15) **break**;
- (16) **end if**
- (17) **if** $Q_q \in U$ **then**
- (18) Q_q removed from U .
- (19) $B_k = B_k \cup Q_q$.
- (20) **end if**
- (21) **end for**
- (22) **end while**
- (23) Compute the centroids c_i via B_n .
- (24) Function $\text{KM}(k, c_i)$
- (25) The final clustering result. $C = \{C_1, C_2, \dots, C_k\}$.

ALGORITHM 1: LSH-based k -means.

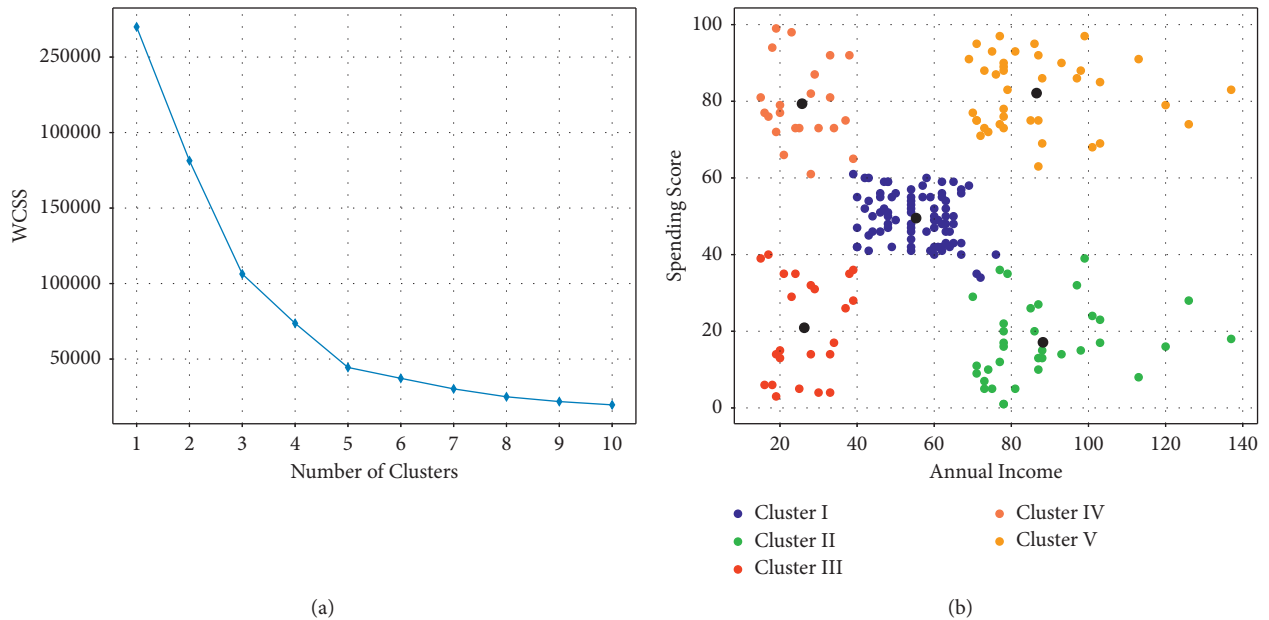


FIGURE 3: Results on the Mall-Customers dataset. (a) Elbow point graph, (b) Results of LSH k -means clustering.

test has been widely used in many fields, especially in algorithm comparison and analysis [40]. It is expressed as follows:

$$R^+ = \sum_{d_j > 0} \text{rank}(d_j) + \frac{1}{2} \sum_{d_j = 0} \text{rank}(d_j), \quad (14)$$

$$R^- = \sum_{d_j < 0} \text{rank}(d_j) + \frac{1}{2} \sum_{d_j = 0} \text{rank}(d_j),$$

where d_j is the difference in clustering performance between the two algorithms on the j -th dataset, and the absolute values of their difference are arranged in the ascending order. If the rank is the same, we take the average value. R^+ implies that the sum of ranks for the algorithm is better than the other, and R^- implies the opposite.

The calculations for the eight aforementioned datasets are presented below.

$$\begin{aligned} R^+ &= 7 + 8 + 2 + 5 + 6 = 28, \\ R^- &= 4 + 1 + 3 = 8. \end{aligned} \quad (15)$$

Let $T = \min(R^+, R^-) = 8$; we get $T = 8$. According to the critical value table of the Wilcoxon test, we can judge that the difference between algorithms is significant under the condition $\alpha = 0.05$. Furthermore, as shown in Table 2, there are five datasets for the LSH-based k -means, which is hence better than the enhanced k -means; thus, in terms of quantity, the LSH-based k -means algorithm outperforms the enhanced k -means algorithm. Therefore, we can judge that the efficiency of LSH-based k -means is significant.

In addition to comparison with actual categories, we further distinguish the clustering effects of k -means clustering and the AC algorithm. A tight and separative indicator is used to evaluate the clustering results [44], which is defined as follows:

$$S(U, k) = \frac{1/n \sum_{i=1}^n \sum_{j=1}^k (u_{ij}) |x_i - c_j|^2}{\min_{p,q=1,2,\dots,k} |c_p - c_q|}, \quad (16)$$

where c_p , c_q , and c_j denote the cluster centers, x_i is any sample in the dataset, k is the number of clusters, and U is the sample set. The Xie–Beni (XB) index [45, 46] is based on intracluster and intercluster distances; it is formulated in terms of the cluster compactness and separation between the clusters. We use the XB index for the evaluation of the cluster effects, and it is defined as follows:

$$\text{XB} = \max\{S(U, k)\}, \quad k = 2, 3, \dots, n-1, \quad (17)$$

where $S(U, k)$ is the ratio of the average distance between data objects and their corresponding clustering centers to the minimum distance of the cluster centers. The smaller the value of $S(U, k)$, the higher is the clustering quality. The results are summarized in Table 3.

From the XB value calculated in Table 3, we can conclude that the difference between the algorithms is significant. The XB value of the AC algorithm is the largest, while that of the LSH-based k -means algorithm is the smallest, which implies

that the LSH k -means algorithm outperforms the other algorithms in the experiment. Thus, the experimental results verify the effectiveness and superiority of the proposed method. Therefore, it can finally be applied to the empirical analysis. In the next section, we describe the application of LSH k -means to business hall analysis.

4.2. Business Hall Analysis. In reality, business hall resource allocation may be unreasonable. For example, some business halls may be busy, while others may be idle. This may be caused by overlapping user coverage in different business halls, unreasonable location of the business halls, and a large proportion of low-value businesses. In this section, we experimentally verify the effectiveness of our index system and analyze the results of the proposed LSH- k -means model.

4.2.1. Business Hall Clustering. When the index system is established as described in Section 3.1, we get the characteristic information of the business hall. After data pre-processing, the number of clusters is determined subjectively. Consider the load intensity and time change information for the business hall. The load intensity can be categorized into High, Medium, and Low, and the load trend can be categorized into No change, Slow growth, Fast growth. Thus, a nine-square grid (Figure 4(b)) map can be obtained. At the same time, by referring to the knowledge of field experts, the number of clusters can be defined as 9 for the subjective clustering methods. After the clusters are determined, the extracted feature indicators can be taken as the input, and the clustering model is implemented. For the LSH k -means algorithm, the distance parameter was selected as the Euclidean distance, the maximum number of iterations was set to 500, the number of seeds was set to 10, and the number of the clusters was set to 9. Then, the outcomes were obtained, as shown in Table 4 and Figure 4(a).

Meanwhile, in the case of different predetermined cluster numbers for the subjective clustering methods, the AC algorithm determined the clustering number automatically, which was computed on the basis of the similarity between the samples. Here, the similarity was set at 95%, and the algorithm was implemented at the same time. Thus, the result was exactly consistent with that of the LSH k -means algorithm. The details are presented in Table 5. For example, the first and sixth samples, the second sample, and the third sample were clustered into the same category.

The final classification results obtained from the model can provide the load grades and decision-making suggestions, which can serve as a basis for site planning optimization of the business halls. In addition, the results of the two algorithms were consistent, which indicate that the LSH k -means algorithm is effective and a stable result was obtained. Accordingly, further optimization action can be implemented.

4.2.2. Optimization Analysis. As shown in Figure 5(a), the 16th and 17th business halls are both in Class I. This category indicates that the current load is Low, and the load trend remains unchanged, implying that the business hall

TABLE 2: Experimental results on UCI datasets.

Datasets	k -means [25] (%)	Enhanced k -means [38] (%)	LSH k -means (this work) (%)	Difference (%)	Rank
Balance	51.62	53.16	54.87	1.71	7
Wine	57.30	65.18	70.22	5.04	8
Breast	93.85	93.99	94.43	0.44	2
Diabet	66.28	67.76	66.72	-1.04	4
Iris	89.33	90.67	89.77	-0.90	3
Hayes-roth	34.09	35.19	35.19	0.00	1
Tic-tac-toe	54.02	56.01	57.17	1.16	5
Bupa	44.64	44.93	46.43	1.50	6

TABLE 3: XB evaluation value.

Algorithms	Iterations similarity	Category			XB
		C_1	C_2	C_2	
k -means [25]	8	27	102	49	0.7067
Enhanced k -means [38]	4	47	69	62	0.7045
AC algorithm [41]	95%	27	126	25	2.1694
LSH k -means (this work)	3	48	68	62	0.7033

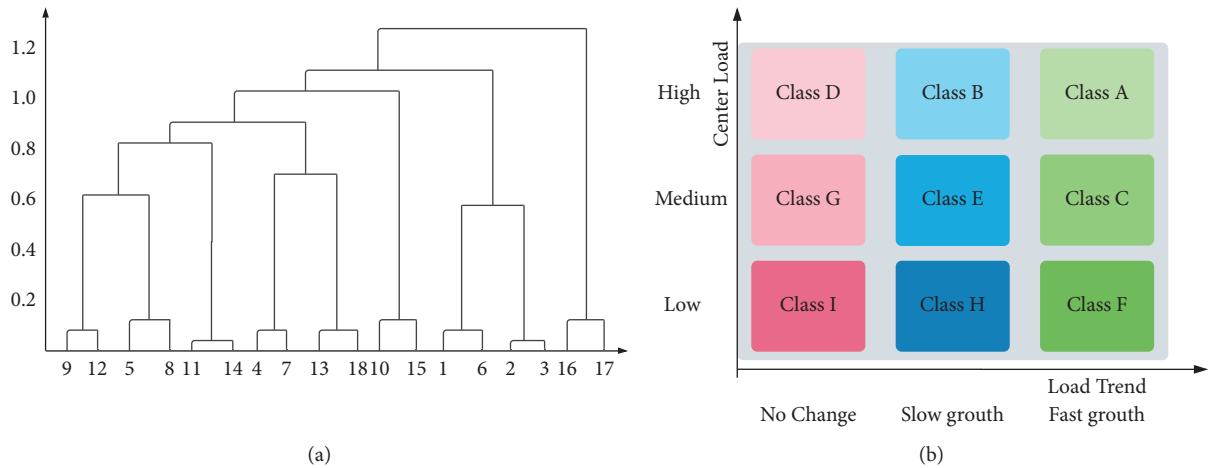


FIGURE 4: Results of LSH k -means clustering on business hall dataset. (a) Diagram of clustering results, (b) Nine-square grid map.

TABLE 4: Clustering result of LSH- k -means.

No.	Full data Attributes	1	2	3	4	5	6	7	8	9
S	0.55	0.25	0.60	0.20	0.10	0.96	0.50	0.95	0.50	0.90
Q	0.00	0.51	-0.51	0.01	-0.50	0.50	0.53	-0.53	0.01	0.01
V	0.62	0.75	0.50	0.45	0.40	0.80	0.60	0.80	0.65	0.70
FH	0.32	0.05	0.08	0.03	0.01	0.95	0.25	0.60	0.08	0.09
FL	0.35	0.85	0.10	0.80	0.95	0.02	0.25	0.10	0.06	0.02
RH	0.37	0.10	0.70	0.80	0.01	0.02	0.50	0.60	0.65	0.03
RL	0.51	0.30	0.10	0.04	0.98	0.90	0.50	0.08	0.80	0.90

resources are redundant in this area. The business halls in this class are idle, and the site may be unreasonable. In addition, the merger of business halls, relocation, and reduction of resource input in this area should be considered.

By contrast, for Class A (the red part of Figure 5(b)), it can be seen that the current load and load trend are both High, which implies that the business volumes of the

business halls are large, and the load trend change is still on the rise. Currently, the first and sixth business halls belong to this category. The future trend is still likely to be growing, and the business volumes will keep increasing. Therefore, this area is where more business hall resources need to be input, and the optimal site planning of business halls should be considered accordingly. We can define the objective

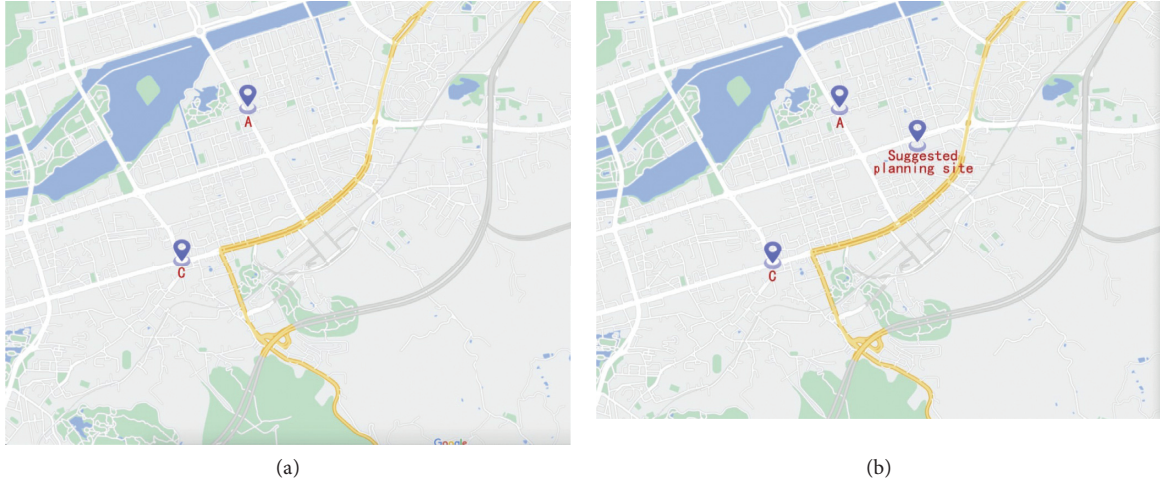


FIGURE 5: Illustration of business hall location optimization. (a) Location of business halls in class A; (b) Suggested location for supplementary business hall.

TABLE 5: Clustering result of the AC algorithm.

Result	S	Q	V	FH	FL	RH	RL
{C1}	0.95	0.5	0.8	0.95	0.01	0.02	0.9
{C2}	0.89	0.01	0.69	0.89	0.02	0.02	0.89
{C2}	0.9	0	0.7	0.9	0.01	0.03	0.9
{C3}	0.94	-0.56	0.79	0.59	0.09	0.59	0.9
{C4}	0.5	0.5	0.6	0.25	0.25	0.5	0.5
{C1}	0.96	0.49	0.79	0.94	0.02	0.01	0.89
...
{C9}	0.09	-0.49	0.39	0.01	0.94	0.01	0.97
{C9}	0.1	-0.5	0.4	0	0.95	0	0.98
{C8}	0.59	-0.51	0.49	0.07	0.09	0.69	0.09

function of the optimal site for the input business hall resources as follows:

$$\text{Obj}_{\min} = \frac{\sum_{k=1}^n F_k}{n} + \sum_{i=1} h_i d(v_i, x), \quad (18)$$

where F_k is the quantitative value of factors that affect the rationality of business hall location, n is the number of factors, $d(v_i, x)$ is the distance function, h_i is the weight, and x is the target point to be solved. Thus, according to the objective function and relevant coordinate information of key units in the area, we can compute the optimal planning location of the business hall using the optimization algorithm. Here, the optimal location of the business hall was computed as [100.0644, 128.8199], and the optimal solution was 527.0368. The details are shown in Figure 5.

The 9th and 12th business halls belong to the Class E, which shows that the current load and load trends are both normal, and the status is stable. Therefore, the business halls in this class are not the current focus of optimization. In addition, the other classes are similar to this category, which is also not the current focus. The main objects are Class I and Class A, that is, excessive or insufficient business hall resources are mainly concentrated in these two classes, which are the focus of our optimization analysis.

5. Conclusion

Excessive or insufficient business hall resources may result in unreasonable resource allocation, which adversely affects the value of an entity business hall. Therefore, proper characteristic parameters are the key factors for analyzing the business hall, which strongly affect the final analysis results. According to the time change and load trend, multiple variables such as average load rate, actual load trend, and high-frequency load are extracted as the characteristic indexes of the business hall. In this study, a characteristic analysis method for the economic operation of a business hall was developed, and the specific calculation process was presented; accordingly, the feature engineering was established. Moreover, based on the load intensity and time change information of business halls, we built an index system and performed further optimization analysis. The key characteristic indicators extracted were the average waiting time, ticket handling time, and business type, and a model for evaluating business hall efficiency was established. The model obtained the load grading of each business hall by the relevant variable input, which provided a basis for optimal site planning of the business halls.

An empirical study showed that the LSH- k -means clustering method outperforms the direct prediction method, provides expected analysis results and decision optimization suggestions for business halls, and serves as a basis for the optimal layout of business halls. In addition, by considering the load intensity and time change information, the cluster number was determined according to the characteristic analysis results, with a certain theoretical and practical significance. In the future, we will explore and develop a general method to automatically determine the parameters and use it in practical applications.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

- [1] M. L. Brandeau and S. S. Chiu, "An overview of representative problems in location research," *Management Science*, vol. 35, no. 6, pp. 645–674, 1989.
- [2] W. Shijun, H. Feilong, and J. Lili, "Locations and their determinants of large-scale commercial sites in changchun," *China*, vol. 70, no. 6, pp. 893–905, 2015.
- [3] F. Goreaud and R. Pélissier, "On explicit formulas of edge effect correction for Ripley's K -function," *Journal of Vegetation Science*, vol. 10, no. 3, pp. 433–438, 1999.
- [4] J. W. Cohen and O. J. Boxma, *Boundary Value Problems in Queueing System Analysis*, Elsevier, Amsterdam, Netherlands, 2000.
- [5] D. R. Anderson, D. J. Sweeney, T. A. Williams, J. D. Camm, and J. J. Cochran, *An Introduction to Management Science: Quantitative Approach*, Cengage learning, Boston, MA, USA, 2018.
- [6] G. Lin, X. Chen, and Y. Liang, "The location of retail stores and street centrality in guangzhou, China," *Applied Geography*, vol. 100, pp. 12–20, 2018.
- [7] S. Kang, "Warehouse location choice: a case study in los angeles, ca," *Journal of Transport Geography*, vol. 88, Article ID 102297, 2020.
- [8] X. Hui, "Optimization model and algorithm research of business hall service channel power," *Electronic Test*, vol. 1, pp. 20–21, 2014.
- [9] X. T. Yan, Y. Zhang, Y. J. Huang, and W. U. Ying-Chun, "Management application and service data integration of the electricity supply business hall," *Power Demand Side Management*, vol. 37, pp. 50–52, 2017.
- [10] B. Baraldi, "A survey of fuzzy clustering algorithms for pattern recognition," *IEEE Transactions on Systems, Man, and Cybernetics. Part B, Cybernetics*, vol. 29, 1999.
- [11] J. Lu, W. Gang, W. Deng, and K. Jia, "Reconstruction-based metric learning for unconstrained face verification," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 1, pp. 79–89, 2014.
- [12] M. Yambal and H. Gupta, "Image segmentation using fuzzy c means clustering: a survey," in *Proceedings of the 2010 6th International Conference on Emerging Technologies (ICET)*, Islamabad, Pakistan, October 2010.
- [13] H. Zhang, T. W. Chow, and Q. M. Wu, "Organizing books and authors by multilayer som," *IEEE Transactions on Neural Networks & Learning Systems*, vol. 27, no. 12, p. 2537, 2015.
- [14] R. C. De Amorim and C. Hennig, "Recovering the number of clusters in data sets with noise features using feature rescaling factors," *Information Sciences*, vol. 324, pp. 126–145, 2015.
- [15] C.-W. Tsai, W.-L. Chen, and M.-C. Chiang, "A modified multiobjective ea-based clustering algorithm with automatic determination of the number of clusters," in *Proceedings of the 2012 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pp. 2833–2838, IEEE, Seoul, Korea, October 2012.
- [16] C. Hennig and T. F. Liao, "How to find an appropriate clustering for mixed-type variables with application to socio-economic stratification," *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, vol. 62, no. 3, pp. 309–369, 2013.
- [17] W. Fu and P. O. Perry, "Estimating the number of clusters using cross-validation," *Journal of Computational & Graphical Statistics*, vol. 29, no. 1, pp. 162–173, 2020.
- [18] S. Vassilvitskii and D. Arthur, "K-means++: the advantages of careful seeding," in *Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 1027–1035, New Orleans LA, USA, January 2006.
- [19] M. A. Masud, J. Z. Huang, C. Wei et al., "I-nice: a new approach for identifying the number of clusters and initial cluster centres," *Information Sciences*, vol. 466, pp. 129–151, 2018.
- [20] M. Erisoglu, N. Calis, and S. Sakallioğlu, "A new algorithm for initial cluster centers in k-means algorithm," *Pattern Recognition Letters*, vol. 32, no. 14, pp. 1701–1705, 2011.
- [21] X. Shen, W. Liu, I. Tsang, F. Shen, and Q.-S. Sun, "Compressed k-means for large-scale clustering," in *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, San Francisco CA USA, February 2017.
- [22] A. Andoni and P. Indyk, "Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions," in *Proceedings of the 2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06)*, pp. 459–468, IEEE, Berkeley, CA, USA, October 2006.
- [23] H. Koga, T. Ishibashi, and T. Watanabe, "Fast agglomerative hierarchical clustering algorithm using locality-sensitive hashing," *Knowledge and Information Systems*, vol. 12, no. 1, pp. 25–53, 2007.
- [24] D. Gorisse, M. Cord, and F. Precioso, "Locality-sensitive hashing for chi2 distance," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 2, pp. 402–409, 2011.
- [25] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the 5th Berkeley symposium on mathematical statistics and probability*, vol. 1, pp. 281–297, Oakland, CA, USA, July 1967.
- [26] X. Wu, V. Kumar, J. Ross Quinlan et al., "Top 10 algorithms in data mining," *Knowledge and Information Systems*, vol. 14, no. 1, pp. 1–37, 2008.
- [27] A. K. Jain, "Data clustering: 50 years beyond k-means," *Pattern Recognition Letters*, vol. 31, no. 8, pp. 651–666, 2010.
- [28] X. Peng, I. W. Tsang, J. T. Zhou, and H. Zhu, "K-meansnet: when k-means meets differentiable programming," <https://arxiv.org/abs/1808.07292>.
- [29] S. Lloyd, "Least squares quantization in pcm," *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–137, 1982.
- [30] J. C. Dunn, "A fuzzy relative of the isodata process and its use in detecting compact well-separated clusters," *Journal of Cybernetics*, vol. 3.
- [31] J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*, Springer Science & Business Media, Berlin, Germany, 2013.
- [32] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," *Kdd*, vol. 96, pp. 226–231, 1996.
- [33] D. Comaniciu and P. Meer, "Mean shift: a robust approach toward feature space analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 5, pp. 603–619, 2002.
- [34] P. Indyk and R. Motwani, "Approximate nearest neighbors: towards removing the curse of dimensionality," in *Proceedings of the 13th Annual ACM Symposium on Theory of Computing*, pp. 604–613, Dallas TX USA, May 1998.

- [35] A. Gionis, P. Indyk, and R. Motwani, "Similarity search in high dimensions via hashing," *Vldb*, vol. 99, pp. 518–529, 1999.
- [36] B. Bahmani, B. Moseley, A. Vattani, R. Kumar, and S. Vassilvitskii, "Scalable k-means++," <https://arxiv.org/abs/1203.6402>.
- [37] W. Hu, Y. Fan, J. Xing, L. Sun, Z. Cai, and S. Maybank, "Deep constrained siamese hash coding network and load-balanced locality-sensitive hashing for near duplicate image detection," *IEEE Transactions on Image Processing*, vol. 27, no. 9, pp. 4452–4464, 2018.
- [38] J. Chen, D. Zhang, and Y. Nanekaran, "Research of power load prediction based on boost clustering," *Soft Computing*, vol. 25, no. 8, pp. 6401–6413, 2021.
- [39] D. J. Newman, "Uci repository of machine learning database," <http://www.ics.uci.edu/mllearn/MLRepository.html>.
- [40] J. Chen, D. Zhang, and Y. A. Nanekaran, "An economic operation analysis method of transformer based on clustering," *IEEE Access*, vol. 7, pp. 127956–127966, 2019.
- [41] F. Lemke and J.-A. Müller, "Self-organising data mining," *Systems Analysis Modelling Simulation*, vol. 43, no. 2, pp. 231–240, 2003.
- [42] R. L. Thorndike, "Who belongs in the family?" *Psychometrika*, vol. 18, no. 4, pp. 267–276, 1953.
- [43] L. Deng, J. Pei, J. Ma, and D. L. Lee, "A rank sum test method for informative gene discovery," in *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 410–419, Seattle, WA, USA, August 2004.
- [44] M. Halkidi, Y. Batistakis, and M. Vazirgiannis, "On clustering validation techniques," *Journal of Intelligent Information Systems*, vol. 17, no. 2, pp. 107–145, 2001.
- [45] X. L. Xie and G. Beni, "A validity measure for fuzzy clustering," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, no. 8, pp. 841–847, 1991.
- [46] M. Singh, R. Bhattacharjee, N. Sharma, and A. Verma, "An improved xie-beni index for cluster validity measure," in *Proceedings of the 2017 Fourth International Conference on Image Information Processing (ICIIP)*, pp. 1–5, IEEE, Shimla, India, December 2017.

Research Article

Quantitative Analysis and Prediction of Global Terrorist Attacks Based on Machine Learning

Xiaohui Pan ^{1,2}

¹*School of Artificial Intelligence and Law, Shanghai University of Political Science and Law, Shanghai 201701, China*

²*School of Management, Shanghai University, Shanghai 200444, China*

Correspondence should be addressed to Xiaohui Pan; panxiaohui@shupl.edu.cn

Received 15 May 2021; Revised 1 September 2021; Accepted 15 September 2021; Published 27 September 2021

Academic Editor: Wei-Chuen Yau

Copyright © 2021 Xiaohui Pan. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Terrorist attacks pose a great threat to global security, and their analysis and prediction are imperative. Considering the high frequency of terrorist attacks and the inherent difficulty in finding related terrorist organizations, we propose a classification framework based on ensemble learning for classifying and predicting terrorist organizations. The framework includes data preprocessing, data splitting, five classifier prediction models, and model evaluation. Based on a quantitative statistical analysis of terrorist organization activities in GTD from 1970 to 2017 and feature selection using the SelectKBest method in scikit learn, we constructed five classification and prediction models of terrorist organizations, namely, decision tree, bagging, random forest, extra tree, and XGBoost, and utilized a 10-fold cross-validation method to verify the performance and stability of the proposed model. Experimental results showed that the five models achieved excellent performance. The XGBoost and random forest models achieved the best accuracies (97.16% and 96.82%, respectively) of predicting 32 terrorist organizations with the highest attack frequencies. The proposed classifier framework is useful for the accurate and efficient prediction of terrorist organizations responsible for attacks and can be extended to predict all terrorist organizations.

1. Introduction

Terrorism is a complex political and social phenomenon. Terrorist attacks have a significant threat to the safety and security of the international community and have become one of the greatest obstacles to the sustainable development of global social security. Antiterrorism is an important part of global security governance, which is a sustainability issue that guarantees global security development. At present, terrorist attacks occur frequently, which leads to significant threats and poses a challenge to global social security governance [1]. According to statistics from the Global Terrorism Database (GTD) [2], more than 200,000 terrorist attacks have been recorded from 1970 to the present day. Terrorist attacks typically involve high lethality and destructive power and directly cause massive casualties and property losses. In addition, they bring tremendous psychological pressure on people. In summary, terrorist attacks result in social unrest to a certain extent, obstructing the

regular order of work and life and thus greatly hindering economic development.

The analysis and prediction of terrorist attacks support targeted attacks on terrorist groups and provide valuable information for antiterrorism and terrorism prevention operations, enabling authorities to find new or hidden terrorists as soon as possible to reduce human and property losses, prevent problems and improve the security and stability of social life. The patterns of attacks planned and carried out by terrorists may seem random on the surface, but in fact, they are typically organized and premeditated actions chosen carefully and deliberately. Moreover, attacks by the same organizations and individuals tend to be substantially related in terms of certain distinguishable characteristics. Therefore, there must be some patterns or informal rules guiding the activities of terrorist organizations. After analyzing these characteristic patterns of activity by terrorist organizations, authorities can make more detailed predictions and analyses of terrorist organizations to

attack them more accurately and increase the time available for the prevention and prediction of terrorist attacks. The GTD provides researchers with comprehensive, reliable, and open-source data, in which there are many potential correlations and patterns to be found. Mining and identifying these patterns using digitally driven methods has become a research topic in the field of informatics.

During the past decades, scholars have established valuable models and algorithms for early warning and prediction of terrorist attacks. Ding et al. [3] demonstrated a novel method using relatively popular and robust machine learning methods to simulate the risk of terrorist attacks at a global scale based on multiple resources, long time series, and globally distributed datasets. The model performed relatively well in predicting the places where terror events might occur in 2015. Chuang et al. [4] studied the spatio-temporal patterns of terrorist attacks by Al Qaeda (AQ), the Islamic State of Iraq and Syria (ISIS), and various local militias or insurgents by applying data-driven, unsupervised k -means clustering to the GTD. Petroff et al. [5] proposed a hidden Markov model to generate early warnings of specific terrorist attacks. Gohar et al. [6] proposed a new collection framework for classifying and predicting terrorist organizations. The framework consists of four basic classifiers, including naive Bayes (NB), k -nearest neighbor (KNN), Iterative Dichotomiser 3 (ID3), and a decision stump (DS). Tolan et al. [7] employed classification techniques to compare five basic classifiers, including naive Bayes, NB, KNN, and support vector machine (SVM), and utilized the GTD to study terrorism and terrorist reactions. Meng et al. [8] proposed an optimized hybrid classifier including data collection, preprocessing, hybrid classification, mining, and classifier testing as a framework for terrorist attack prediction. Bu et al. [9] combined an SVM with an intelligent tuned harmony search (ITHS) algorithm to build an ITHS-SVM model for terrorist attack classification, which provides learning and curve-fitting functions while optimizing SVM parameters. Li et al. [10] proposed a comprehensive framework that combined social network analysis, wavelet transform, and pattern recognition approaches to investigate the dynamics and eventually predict the attack behavior of terrorist groups. Hu et al. [11] developed a risk assessment system for terrorist attacks through a quantitative analysis of the GTD and clustered and ranked terrorist attacks according to the results of terrorist attack rating models. Campedelli et al. [12] proposed the use of temporal meta-graphs and deep learning to forecast future terrorist targets using real data of attacks in Afghanistan and Iraq from 2001 to 2018. The experimental results showed that bidirectional LSTM networks achieve superior forecasting performance compared to other algorithms. Although these existing models and algorithms can be used to classify and predict terrorist activities, their accuracy remains less than ideal, and the coverage of the number of terrorist organizations is insufficiently comprehensive, with some covering only a few years or a few regions of a country.

In addition, methods designed to perform analysis and prediction of terrorist attacks have been gradually developed from the perspective of network science. Campedelli [13]

proposed a new methodological framework integrating network science, mathematical modeling, and deep learning to compare and analyze the world's most active jihadist terrorist organizations (i.e., the Islamic State, the Taliban, AQ, Boko Haram, and Al-Shabaab), investigate their behavioral patterns, and forecast their future actions. Campedelli et al. [14] first built a multiparty network including information about terrorist organizations and tactics, weapons, targets, and active areas by using GTD data from 1997 to 2016. Then, a new clustering algorithm was proposed, which used von Neumann entropy for modal weighting. Compared with the other two clustering methods, the experimental results showed that the entropy-based method tended to reliably reflect the data structure naturally generated by the baseline Gower method. Desmarais and Cranmer [15] constructed a network of transnational terrorist attacks in which the source (sender) and target (receiver) countries share a directed edge. A deterministic, similarity-based link prediction framework was integrated into a probabilistic modeling approach to develop an edge-forecasting method. Experiments showed that probabilistic link prediction could not only accurately predict terrorist actions but showed promise to predict the onset of terrorist hostilities between a source and a target.

In recent years, artificial intelligence has rapidly emerged and gradually matured, empowering scientific and technological products, and promoting its development in human society. Artificial intelligence (machine learning) focuses on data training and fitting. With the support of a large amount of raw data, models can achieve a high prediction accuracy. Specific patterns may be noted in the organization, planning, and development of terrorist attacks, and the accumulated terrorist attacks over the years provide a large amount of characteristic data. Therefore, artificial intelligence is expected to become an excellent tool for analyzing and predicting the rules of terrorist attacks. Guo et al. [16] summarized three ways to improve conflict forecasting and called on the UN to invest in data-driven predictive methods for promoting peace. Three new methods were developed, including new machine learning techniques, more information on the wider causes of conflicts and their resolution, and theoretical models that better reflect the complexity of social interactions and human decision-making.

However, well-known machine learning problems, such as data imbalance, the curse of dimensionality, and false correlation, may cause machine learning algorithms to be inaccurate. When using datasets with a sufficient number of labeled cases, machine learning can help police departments detect local crimes and predict when and where crimes will occur. However, when predicting the identity of offenders or criminals, especially in terrorist attacks, the number of false positives and false negatives can be relatively high when implementing these algorithms because the feature attributes are numerous and redundant.

The present work is focused on the application of machine learning to study the characteristics of terrorist attacks, quantitative analysis, and prediction. There are many types of artificial intelligence models and methods, and the

ensemble learning method in machine learning has higher accuracy and better generalization ability. In this study, we perform a quantitative analysis of the activities of terrorist organizations in GTD from 1970 to 2017 and propose a classification framework based on ensemble learning to accurately classify and predict terrorist organizations. First, we perform a quantitative analysis of global terrorist attacks and study the varying attributes and characteristics of different times, places, and terrorist organizations in the GTD. Second, we perform essential data preprocessing, including data cleaning (such as missing value processing and data conversion), feature selection, and data splitting. Third, we construct a classification framework based on ensemble learning for classifying and predicting terrorist organizations. The framework incorporates five current mainstream models: decision tree, bagging, random forest, extra tree, and XGBoost. Finally, we conduct comprehensive experiments to evaluate the performance of these algorithms through a set of metrics and provide a visual exploratory discussion. Experimental results show that the XGBoost and the random forest models were the most effective and achieved the highest accuracy.

2. Analysis of GTD Dataset

2.1. Quantitative Analysis of GTD Dataset. The dataset was derived from data on terrorist attacks from 1970 to 2017 from the GTD [2] (<https://www.start.umd.edu/gtd/>), which is managed by the National Consortium for the Study of Terrorism and Responses to Terrorism (START). The GTD dataset is considered to be the most comprehensive database for recording global terrorist activity. The information of terrorist organizations in GTD is represented by the “gname,” “gname2,” and “gname3” attribute fields, which, respectively, represent up to three organizations participating in an event. Most events have only a gname field value, and some events may only be represented by an unknown. Therefore, this article is focused on analyzing and predicting the “gname” attribute field (that is, the major organization). For a very small number of events with more than one terrorist organization, we focus on the major terrorist organization. According to the analysis of the dataset, there were 3,537 nonrepeated statistics on the attribute fields of the terrorist organizations recorded. Except for the records of unknown terrorist groups as “gname = Unknown,” there were 3,536 terrorist groups in the dataset.

Our preliminary analysis shows that there were 181,691 identified terrorist incidents in the GTD dataset from 1970 to 2017, excluding incidents with unknown terrorist organizations. Among the cases where terrorist groups have been identified, some terrorist organizations were very active and launched numerous attacks; 19 large terrorist organizations that launched more than 1,000 terrorist attacks, 32 terrorist organizations exceeded 500, and 122 terrorist organizations exceeded 100. These 122 terrorist organizations launched 78,107 terrorist attacks, accounting for more than 79% of all known terrorist group incidents. The number of terrorist attacks, the corresponding number of terrorist

organizations, and the sum of terrorist attacks by these terrorist organizations are listed in Table 1.

The detailed statistics of terrorist attacks and the corresponding number of terrorist organizations are shown in Figure 1. Among them, the areas within the rectangles of each specified range were mutually exclusive.

2.2. Visual Analysis of Terrorist Organizations. The activities of the 3,536 terrorist organizations were further analyzed. To facilitate visual observation, we conducted a detailed analysis of 32 terrorist organizations that carried out more than 500 terrorist attacks from 1970 to 2017. The names, number of attacks, and ranking of these 32 terrorist organizations are shown in Figure 2. The organization with the maximum number of terrorist attacks was the Taliban, which conducted a total of 7,478 terrorist attacks, followed by the Islamic State of Iraq and the Levant (ISIL) and Shining Path (SL) with 5,613 and 4,555 terrorist attacks, respectively. In addition, the number of terrorist attacks slowly decreased from around 3,000 (FMLN) to around 500 (Fulani extremists).

The annual activity distribution of the top 10 terrorist organizations was calculated, as shown in Figure 3. Notably, the overall number of terrorist attacks has shown a significant upward trend in recent years. In particular, the three terrorist organizations ISIL (purple line), Taliban (light blue line), and Al-Shabaab (blue line) planned and carried out terrorist attacks extremely frequently. It is worth noting that ISIL emerged only in 2012 and has since launched more than 1,000 terrorist attacks every year. The intrinsic reason is worthy of in-depth analysis by researchers.

3. Research Methods

In this paper, we propose a classification framework based on ensemble learning to classify and predict terrorist organizations. The framework involved four steps, including data preprocessing, data splitting, construction and training of several ensemble learning classifier models, and classifier model testing, as shown in Figure 4.

3.1. Machine Learning Prediction Model. Machine learning methods are usually divided into supervised and unsupervised approaches. Within the former category, many applications aim at predicting a target variable. The specific method involves establishing a corresponding relationship between the attribute variables and the target variables in the sample dataset, and this mapping relationship is formed by constructing a model from the training dataset. The prediction and evaluation are performed on the testing dataset, and the value of the predicted target variables is then compared with the value of real target variables to derive the prediction accuracy.

For a given terrorist attack, the classification models identify terrorist organizations or individuals in a terrorist attack based on known attribute fields. In the process of supervised machine learning, the existing terrorist event feature data are sent to the classification algorithm model for

TABLE 1: Statistics on the number of terrorist attacks and terrorist organizations.

Number of terrorist attacks	Number of terrorist attack organizations	Sum of terrorist attacks
≥ 1000	19	50200
≥ 500	32	58520
≥ 100	122	78107
≥ 50	210	84339
≥ 5	936	94871

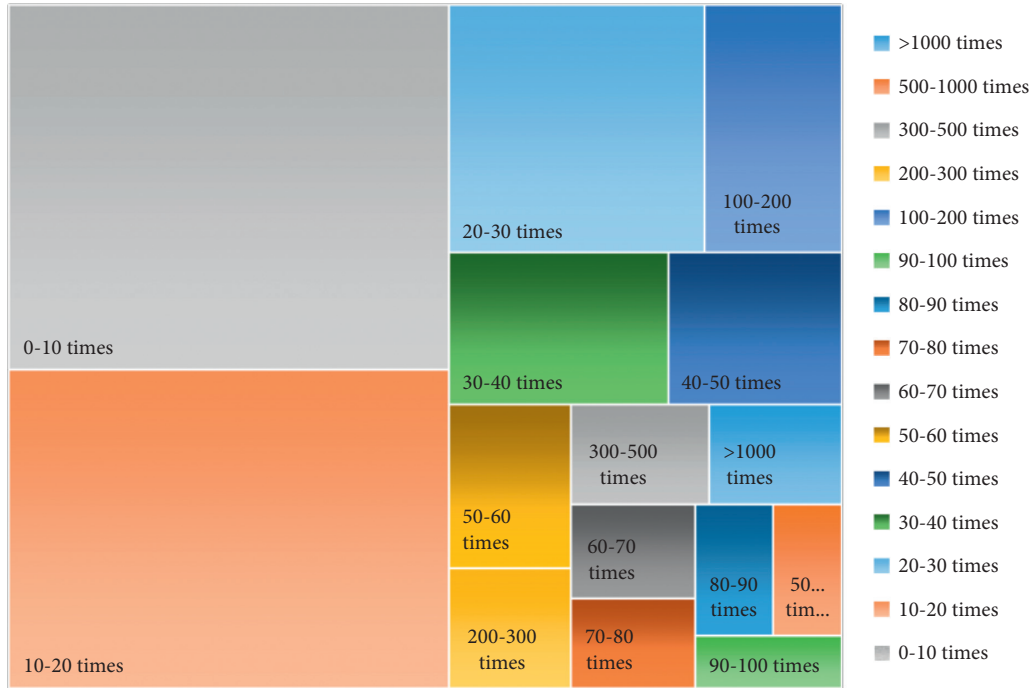


FIGURE 1: The statistics on the number of terrorist organizations by the number of attacks.

training and learning. Then, the trained model is used to classify the test or new data to predict candidate terrorist organizations or individuals. Therefore, the prediction of terrorist organizations is a multiclassification problem. The primary purpose of this research is to construct classification models for multiclassification tasks. In this study, we used five supervised machine learning classifiers [17] to predict terrorist organizations responsible for various attacks, including decision tree, bootstrap aggregating, random forest, extra trees, and super gradient boost.

Decision tree (DT) algorithms [18, 19] can be used as a supervised learning method. By creating a tree model to learn simple decision rules from data features to predict the value of a target variable, the DT model begins the decision from the root node, and the leaf nodes represent a successful guess or correct prediction. There are three major algorithms for creating DTs: ID3, C4.5, and Classification and Regression Tree (CART). ID3 starts from the root node of the tree and uses information gain to select features to build child nodes. C4.5 uses the information gain ratio to select features, which is regarded as an improvement of ID3. However, these two algorithms cause the problem of overfitting, which requires pruning. The pruning of the DT

removes unnecessary classification features by optimizing the loss function and reducing the overall complexity of the model. CART [20, 21] adopts the Gini index minimization principle to create a tree. It cuts out some subtrees from the bottom of a fully grown DT, making the model simpler. We used CART to create decision trees in this study.

The following four models are considered ensemble learning [22], which is a branch of machine learning. The basic unit of these four models is a decision tree.

Bootstrap aggregating (Bagging) [23, 24] is a classification algorithm that uses a combination strategy. It first obtains m sample sets by extracting the original dataset m times with replacement and then uses each sample set to train m base classifiers separately. Finally, an integrated classifier was constructed by applying a combination strategy to the base classifiers.

Random forest (RF) [25] is an algorithm that integrates multiple DTs through ensemble learning. RF usually uses the mean or mode of the prediction results of each DT in the decision tree set as the final prediction value. The RF in the scikit-learn Python package uses the mean as a predictor. Compared with a single DT, RF is less likely to be affected by overfitting because each DT of the random forest cannot see

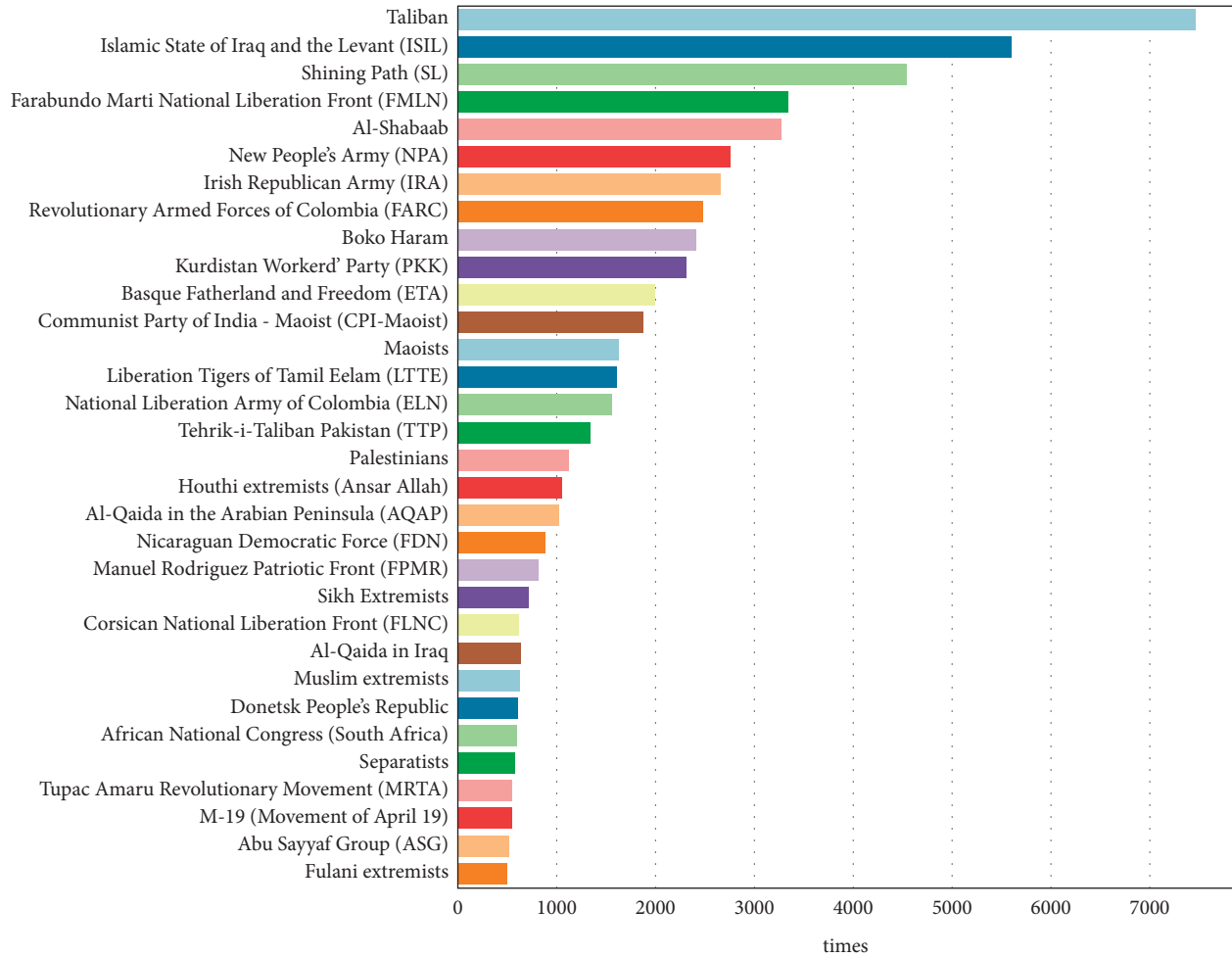


FIGURE 2: Rankings of terrorist organizations with more than 500 terrorist attacks.

the full view of the training set. Each DT only trained a part of the attribute data and did not remember all the noise of the training set.

Extra trees (ET) [26, 27] are also composed of many DTs, such as RF. These decision trees use random features and random thresholds for the node division. ET provides additional randomness, which suppresses overfitting but also increases the bias to some extent. The difference between ET and RF is that RF uses bagging for random sampling, whereas ET uses all samples. RF finds the optimal attributes based on information entropy and the Gini index in a random subset, while ET finds an eigenvalue entirely at random to divide.

The super gradient boost (XGBoost) [28, 29] is also a classification algorithm that integrates multiple decision trees. It pays more attention to the samples that were learned incorrectly in the previous round during training and makes some improvements on Gradient Boosting by introducing second-order derivatives and approximating the loss function with first- and second-order derivatives so that there is more information in the optimization process. In addition, XGBoost adds a regular term to the loss function to weigh the complexity of the model, making it simpler and preventing overfitting. Compared with the RF, there is no

dependency relationship between the decision trees in the RF, and they can be parallel. However, XGBoost trees are dependent and must be serialized. This model maximizes the integration speed and efficiency of trees and is a very effective integration algorithm.

3.2. *Evaluation Metrics.* After the machine learning classification model for this problem was designed and constructed, it was necessary to evaluate the performance of a classifier to determine the accuracy of a classifier in predicting the class labels of terrorist organizations.

In machine learning, a multiclass classification problem can usually be converted into multiple binary classification problems. Each binary classification problem classifies a group of target objects into one class (i.e., category) and the remaining target objects into another class. The confusion matrix is an analysis table that summarizes the prediction results and the real results in binary classification and multiclass classification [30], as shown in Table 2.

Based on confusion matrices, four commonly used metrics are generally applied to evaluate the performance of machine learning, including accuracy, precision, recall, and *F1* score.

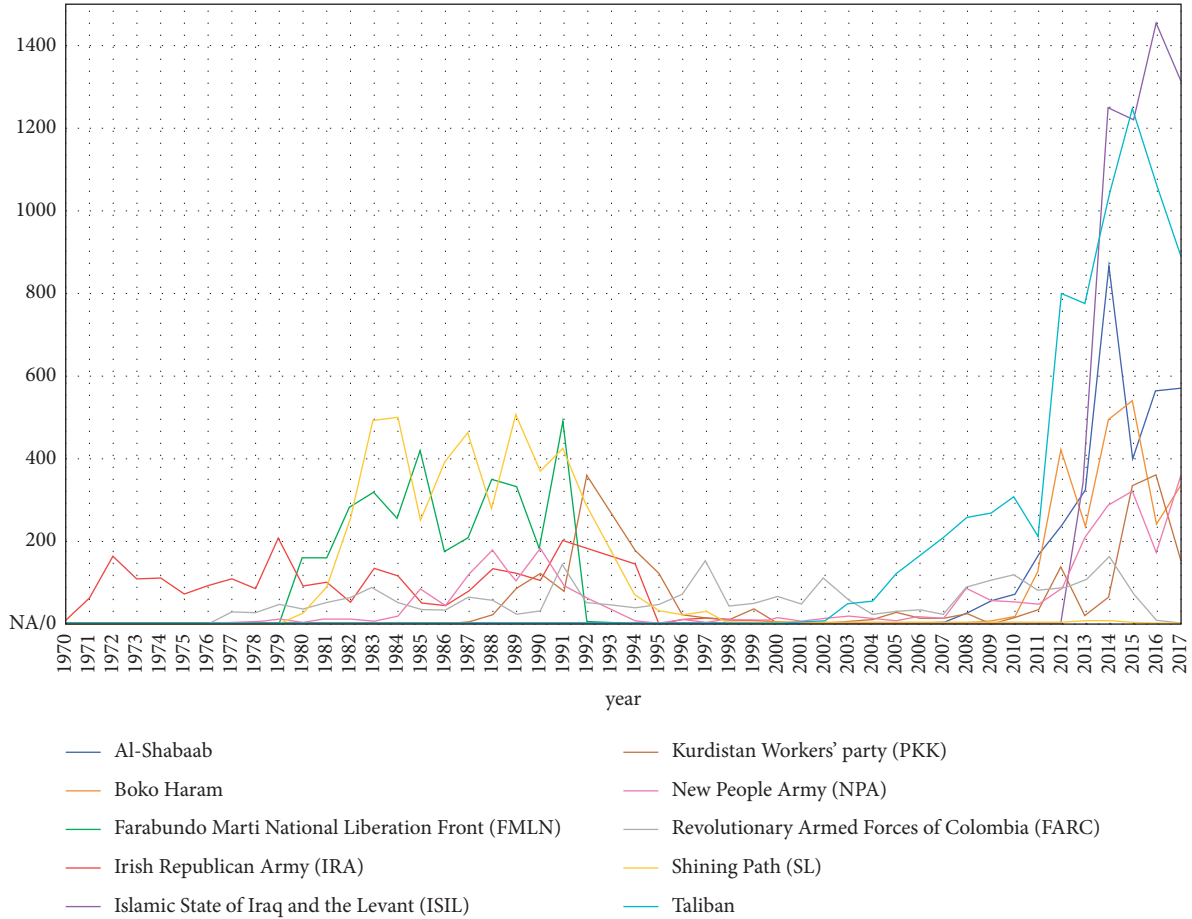


FIGURE 3: The annual activity distribution of the top 10 terrorist organizations.

Among them, accuracy can intuitively reflect the prediction result. Precision and recall are good complementary indicators when accuracy is not sufficient to reflect the detail of the assessment results. The $F1$ score is the harmonic mean value of the precision and recall. According to Table 2, the four metrics are defined as follows:

$$\begin{aligned}
 \text{Accuracy} &= \frac{TP + TN}{TP + TN + FP + FN}, \\
 \text{Precision} &= \frac{TP}{TP + FP}, \\
 \text{Recall} &= \frac{TP}{TP + FN}, \\
 F1 &= \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}.
 \end{aligned} \tag{1}$$

Accuracy is defined as the ratio of correctly predicted samples to the total number of samples. It is the percentage of terrorist organizations correctly classified in an attack. Precision is the ratio of true positive samples among all samples predicted as positive samples. Recall is the ratio of the number of positive samples predicted to the total number of all positive samples. For specific terrorist organization i , precision

(abbreviated as P) refers to the ratio of the number of samples correctly predicted to be terrorist organization i to the number of all samples predicted to be terrorist organization i . Recall (R) refers to the ratio of the number of samples correctly predicted to be terrorist organization i to the number of true samples of a terrorist organization i .

P and R indicators are important to provide an alternative perspective on the false positive rate versus the false negative rate of that terrorist organization. A false negative means that the organization launched an attack that was not correctly identified and instead confused with another terrorist organization. A false positive means that an attack that the group did not initiate was incorrectly identified as its act. Comparing the importance of false positives and false negatives for a terrorist organization is different for different terrorist organizations. Because of the impact this can have on the subsequent handling strategy of a terrorist attack situation, for larger terrorist organizations with more aggressive activities and more severe tactics, false negatives are more important and have more serious consequences, as they may result in negligence and trivialization of the handling strategy. For less aggressive, relatively civilized terrorist organizations, false positives are more important and may likewise result in negligent and dismissive treatment strategies.

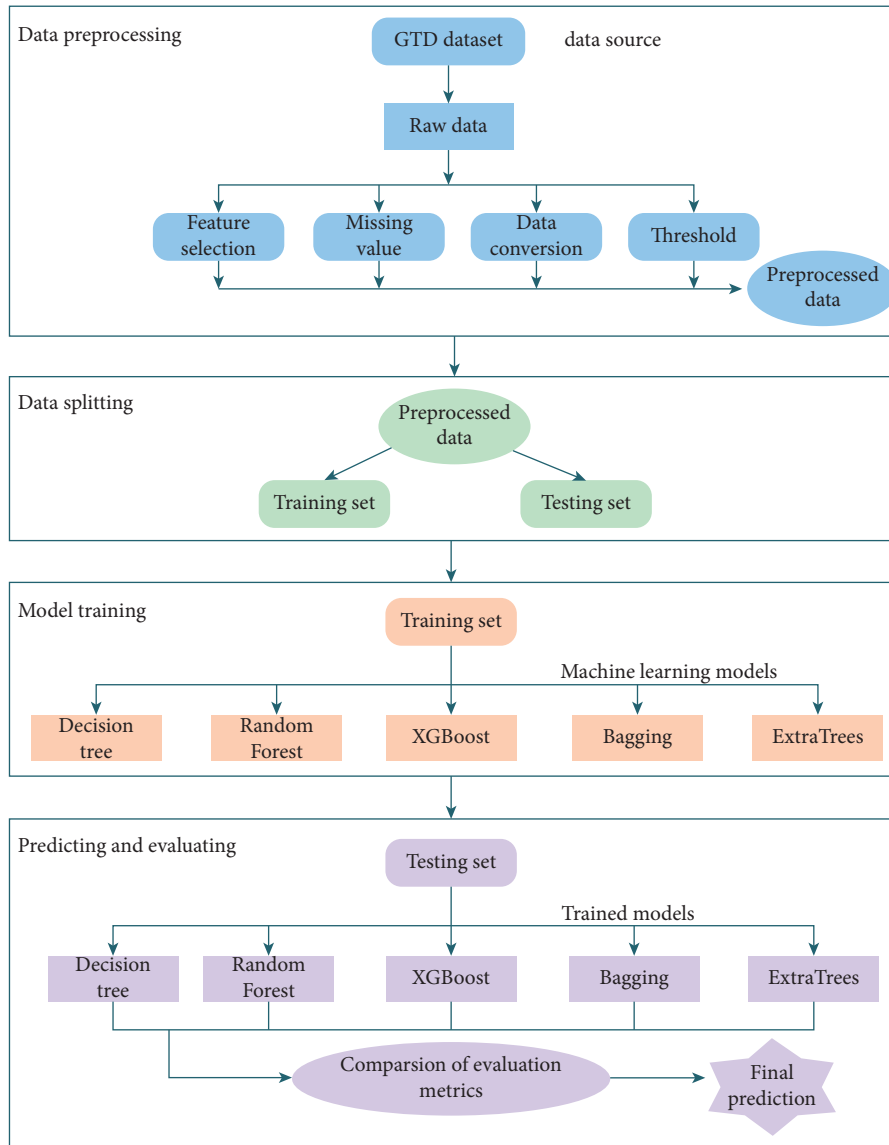


FIGURE 4: The framework for classifying and predicting terrorist organizations.

TABLE 2: Confusion matrix.

Actual category	Predicted category	
	True	False
True	True positive	False negative
False	False positive	True negative

In the evaluation of multiclass classification, P , R , and $F1$ scores were used to evaluate each category. The overall evaluation of all multiple categories is usually performed using accuracy and macroaverage. Macroaverage first calculates the statistical indicators precision, recall, and $F1$ for each category separately and then calculates the arithmetic average for all categories.

4. Experiment Methodology and Result Analysis

Quantitative analysis and modeling prediction of global terrorist attacks were performed in Python 3.6, running on a platform with an Intel Core i7 processor and 24.00 GB DDR RAM. We utilized the Python libraries pandas-0.25.2, numpy-1.17.2, xgboost-1.0.0, and scikit-learn-0.21.3 [31].

For visualization of the analysis results, we used seaborn-0.9.0 and matplotlib- 3.1.1 in Python.

4.1. Data Structure Analysis. The data primarily contained the following attributes of information: GTD serial number, date, event description information, time, location, attack description information, weapon information, target information, victim information, casualty information, and action results. There were many fields under each type of information to enrich the data. Each terrorist attack was stored as a record (i.e., a row) of 137 attributes such as country, year, number of deaths and injuries, and use of weapons. Among them, there were 46 attributes with a completeness of more than 70%.

4.2. Data Preprocessing. In the dataset, the average number of attacks by all terrorist organizations was 28. However, 3,430 terrorist organizations (91% of all terrorist organizations) launched fewer than 28 terrorist attacks, and 2,600 terrorist organizations (73% of all organizations) launched fewer than five terrorist attacks. These 2,600 terrorist organizations launched 4,038 terrorist attacks, which accounted for only 4% of the identified terrorist attacks (i.e., attacks by identified terrorist organizations). If all terrorist organizations were predicted, too many categories and low-sample categories may cause unfavorable training interference noise. Therefore, to make the experiment closer to reality and the trained model more effective, samples with fewer than five terrorist attacks were removed in this study.

Some attributes are unrelated to the prediction of terrorist organizations. Training on these attributes would not only increase the required training time but also render the training results unreasonable or impractical; therefore, data preprocessing operations are essential. At this stage, the GTD dataset was processed through data cleaning, feature engineering, and data normalization.

4.2.1. Data Cleaning. Data cleaning aims to reduce the dimensions of the GTD dataset by detecting and deleting irrelevant or redundant attributes and case records.

First, attribute fields that contained descriptive text or too many missing values (the missing threshold was set to 30%) were removed. Second, missing values in specific attribute fields were filled with the numerical value corresponding to “unknown” according to the data description rules provided by the GTD. Third, some attribute fields were converted into numerical values to facilitate later processing. For example, the “related” attribute field provides the “eventid” of other terrorist attacks’ related to this terrorist attack in text format, and we convert it to the count of related terrorist attacks. The number of event records after these three steps was reduced to 98,909. Fourth, after deleting the records of terrorist attacks with fewer than five terrorist attacks, we filtered the remaining records of terrorist attacks according to five conditions (i.e., ≥ 5 times, ≥ 50 times, ≥ 100 times, ≥ 500 times, ≥ 1000 times). Eventually, the number of

records in the experimental dataset was reduced to 94871 after the data cleaning process.

4.2.2. Feature Engineering. We tended to retain the objective attributes of the terrorist attacks in the GTD and ignored some subjective judgment criteria and some text columns for interpretation and apparently irrelevant attributes, such as crit1-3, country_txt, region_txt, and eventID. Therefore, 45 possible related attributes were left for analysis. Further selections were then made. First, 34 numerical data (int, float) were selected without special processing. Then, the target/victim nationality (natlty1-3) was transformed into int numerical type and was selected. In this way, 37 candidate feature attributes were filtered out. For these 37 properties, it is difficult to determine the feature attributes that should be retained or removed because the remaining feature attributes after data cleaning are correlated in some way. After considering some solution strategies, we used the feature selection function (i.e., SelectKBest) in sklearn to make the selection, and very few adjustments were made.

Based on the above strategies, to simplify the model and improve the prediction accuracy, we selected 36 features for the experiment, including iyear, imonth, iday, extended, country, region, successful attack, suicide attack, attack type1-3, target type1-3, target subtype1-3, target nationality (natlty1-3), weapon type1-4, weapon subtype1-4, property, ishostkid, ransom, related, INT_IDEO, INT_LOG, INT_MISC, and INT_ANY. We used the ExtraTrees classifier to build a forest to rank the importance of the 36 feature attributes, as shown in Figure 5. It may be observed that the three attributes, namely, the country and region where the terrorist attack occurred and the target nationality were the most critical attributes for predicting terrorist organizations.

Thus, the GTD was transformed into a new dataset with a scale of $94871 * 37$ after data preprocessing. Among them, “gname” is the target attribute for prediction, and the remaining 36 attributes are the explanatory features for prediction.

4.3. Data Splitting. In machine learning, the sample dataset is usually partitioned into a testing set and a training set in proportion. Because the classification of the target feature attributes in the dataset is usually unevenly distributed, the training and testing sets are divided according to the proportion of the target features in the sample dataset, such that the proportion of the data in each category of the training set and the testing set is consistent with the proportion of the sample dataset, thereby reducing the misleading predictions of the trained models. The following two methods are generally used in data splitting.

4.3.1. Hold Out. Directly partition the data (Data) into two mutually exclusive sets, one of which is used as the training set (Training) and the other as the testing set (Testing), $Data = Training \cup Test$, $Training \cap Testing = \emptyset$. When dividing the data into a training set and a testing set, the data

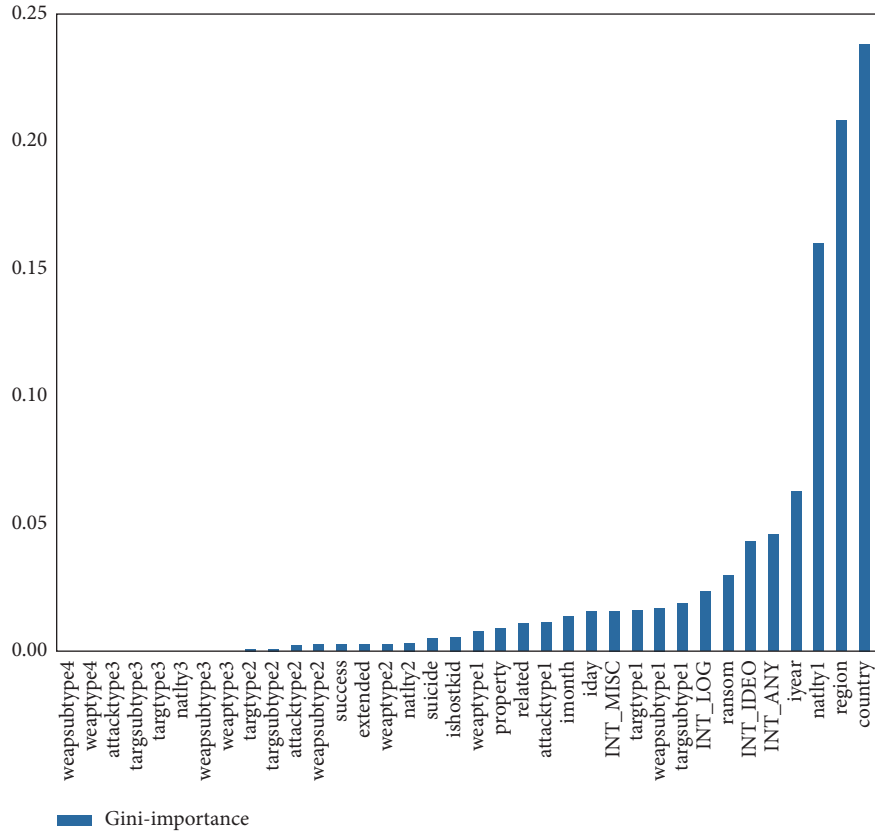


FIGURE 5: Importance ranking of 36 feature attributes.

consistency must be maintained as much as possible to avoid affecting the final result from deviations in the data division process.

4.3.2. *K-Fold Cross Validation.* Divide the data into k mutually exclusive subsets of similar size, namely, $\text{Data} = D_1 \cup D_2 \cup \dots \cup D_k$, $D_i \cap D_j = \emptyset (i \neq j)$. Each time, the union of $k-1$ subsets was used as the training set, and the remaining subset was used as the testing set. Thus, k combinations of the testing and training sets were obtained. The k -fold cross-validation method is a common method used to alleviate the problem of data imbalance.

In this study, the hold-out method was used to split the sample dataset into 90% and 10% portions as the training and testing sets, respectively. The training set was used to build the model, and the testing set was used to test and evaluate the effectiveness of the model. In addition, we used the 10-fold cross-validation method to evaluate the models further and compared the results with those of the hold-out method to verify the stability of the model.

4.4. Model Construction and Evaluation Analysis

4.4.1. Classifier Modeling. In this study, five mainstream classifiers, including DT, Bagging, RF, ET, and XGBoost, were used to classify and predict terrorist attack organizations that perpetrated specific attacks. We first optimized

these models and then evaluated and compared the performances of these models.

Hyperparameters [32] in the machine learning models can be manually set and continuously optimized by trial-and-error minimization. Table 3 briefly introduces the major parameters used in the ensemble learning models.

We optimized these hyperparameters by performing grid searches or random searches for manual tuning of the model and then comparing the accuracy and selecting the optimal parameter value for model performance. After tuning, we obtained the optimal parameters of these five models as follows: The DT uses the CART algorithm by default, and the parameter `random_state` is 42. Bagging uses the KNN classifier as the base classifier, and the `random_state`, `max_features`, and `max_samples` are 30, 0.5, and 0.5, respectively. The RF parameters `n_estimators`, `max_features`, and `random_state` are 500, “sqrt,” and 42, respectively. The ET parameters `n_estimators`, `min_samples_split`, and `random_state` are 10, 2, and 12, respectively. The XGBoost parameter `n_estimators` is 300. All other parameters of the above models were set to default values.

We performed the following two experiments using the hold-out method and the 10-fold cross-validation method and compared the results to verify the performance and stability of the models.

- (1) Hold-out method: the new data obtained after data preprocessing is split into a training set containing

TABLE 3: Description of parameters in ensemble learning models.

Parameter	Description
<i>n_estimators</i>	Number of base classifiers
<i>random_state</i>	Number of seeds of random number generator
<i>max_features</i>	Maximum number of features involved in judgment when splitting nodes
<i>max_samples</i>	Maximum number of samples
<i>min_samples_split</i>	Minimum number of samples required for splitting

90% data and a testing set containing 10% data using the “hold-out” method, and the above five classification models are trained and tested, respectively, to obtain the corresponding accuracy, precision, recall and *F1*.

- (2) 10-fold cross-validation method: owing to the imbalance of terrorist organization type data, the 10-fold cross-validation method is a common method to alleviate the problem of data imbalance. For instance, some terrorist organizations performed a large number of terrorist attacks, while other terrorist organizations performed a small number of attacks. Therefore, the five algorithms were trained ten times through the 10-fold cross-validation method, the output of each training result was retained, and the average value of the ten output results was calculated to obtain the cross-validation accuracy.

4.4.2. Result Evaluation and Analysis. To comprehensively reflect the global prediction performance, we performed predictions in each of the five ranges of terror attack frequency, and the experimental results of the model performance experiment in terms of accuracy, precision, recall, and *F1* indicators are shown in Table 4 and Figure 6.

It may be clearly seen from Figure 6 that the 19 terrorist organizations with the highest terror attack frequency (≥ 1000) had the highest prediction accuracy for all algorithms, and the prediction accuracy of terrorist organizations decreased as the frequency of terror attacks decreased. For 936 terrorist organizations with more than five terrorist attacks, the prediction accuracy was reduced to approximately 0.85. This indicates that as the number of predicted target classes for the multiclassification problem increased substantially, the algorithm’s ability to discriminate some features decreased, and it tended to confuse the target classes with high similarity. From the comparison of the five algorithms, it may be seen that the XGBoost model performs best in the case of the number of terrorist attacks (≥ 1000 , ≥ 500), and the RF model performs best in the case of the number of terrorist attacks (≥ 100 , ≥ 50 , ≥ 5).

To explore the analysis results visually, we focused on terrorist organizations with frequent attacks. Here, we analyze in detail the experimental results of 32 terrorist organizations with no less than 500 terrorist attacks frequency. The experimental results showed that the prediction accuracy of the XGBoost model reached 97.1634%, with a precision of 95.7246%, and the comprehensive evaluation index *F1* was 95.0011%. Random Forests was closely followed, with an accuracy of 96.8216%. We observed that the three models

achieved the highest accuracy among all five models, that is, XGBoost, Random Forests, and ExtraTrees. These models are all ensemble learning algorithms based on the tree model; therefore, it is concluded that the tree model’s ensemble learning classifier was optimal for this research.

To compare the performance of the models before and after alleviating the imbalance of the types of terrorist organization data, the five models were trained and tested using the hold-out method (without alleviating data imbalance) and the 10-fold cross-validation method (with alleviating data imbalance), respectively. The accuracies of the 32 terrorist organizations are shown in Table 5 and Figure 7.

Although the precision of the 10-fold cross-validation method was slightly lower, the accuracy, recall, and *F1* were almost the same. The 10-fold cross-validation method can alleviate data imbalance and avoid misleading models; thus, the models reflect more realistic and effective results. The model metric difference between the two methods is very small, which also shows that the constructed prediction models are relatively stable and accurate.

5. Visual Exploration and Discussion

To observe the effect of the models visually, we used the confusion matrix of 32 terrorist organizations to show the prediction results. In confusion matrices, we can observe the number of correct and incorrect predictions and the results of incorrect predictions (e.g., terrorist organization A is predicted to be terrorist organization B).

Because the XGBoost model outperforms all other models (as shown in Table 5), it is meaningful to explore and analyze the effect of the XGBoost model. The visualization of the confusion matrix for the prediction results of the XGBoost model is shown in Figure 8. Other models can also be visually analyzed in the same manner.

There are 5,852 records (i.e., terrorist attacks) in the test data after data splitting. We draw the confusion matrix of the XGBoost model for 32 terrorist organizations with more than 500 terrorist attacks, as shown in Figure 8. The rows of the confusion matrix represent the actual terrorist organizations in the test data, and the columns of the confusion matrix represent the predicted terrorist organizations in the test data. Therefore, the diagonal values in the confusion matrix represent the number of terrorist attacks in which the corresponding terrorist organization was predicted correctly, while the values outside the diagonal represent the number of corresponding terrorist organizations that were incorrectly predicted as being responsible for attacks. For a model that achieves 100% accurate prediction, the confusion

TABLE 4: Comparison of 5 algorithms for predicting terrorist groups with different attack frequencies.

Summary of data records		Number of terrorist attacks (range)	≥1000	≥500	≥100	≥50	≥5
		Number of terrorist organizations	19	32	122	210	936
		Total number of terrorist attacks	50200	58520	78107	84339	94871
Accuracy	Decision trees		0.982669	0.958647	0.878377	0.854636	0.796164
	Bagging		0.960757	0.931989	0.833312	0.799858	0.740620
	Random forests		0.983068	0.968216	0.904494	0.881195	0.835687
	ExtraTrees		0.979283	0.959501	0.886698	0.860327	0.803225
	XGBoost		0.983466	0.971634	0.853924	0.791439	0.698567
Precision	Decision trees		0.976950	0.928242	0.787521	0.745648	0.478754
	Bagging		0.945956	0.932152	0.771253	0.685609	0.347113
	Random forests		0.979232	0.957727	0.847559	0.817384	0.520747
	ExtraTrees		0.973327	0.942159	0.811242	0.761273	0.476816
	XGBoost		0.978406	0.957246	0.752235	0.523490	0.126893
Recall	Decision trees		0.976073	0.929554	0.786034	0.737920	0.512161
	Bagging		0.940090	0.858106	0.605144	0.523339	0.304769
	Random forests		0.974743	0.934140	0.785265	0.739619	0.511993
	ExtraTrees		0.970025	0.926234	0.761510	0.708550	0.469377
	XGBoost		0.976059	0.944904	0.746525	0.537858	0.138689
F1 score	Decision trees		0.976497	0.928603	0.784185	0.736470	0.481310
	Bagging		0.941057	0.875151	0.633059	0.551191	0.305712
	Random forests		0.976587	0.942883	0.805488	0.754597	0.502975
	ExtraTrees		0.971609	0.932798	0.779096	0.723834	0.459432
	XGBoost		0.977118	0.950011	0.745925	0.523800	0.130349

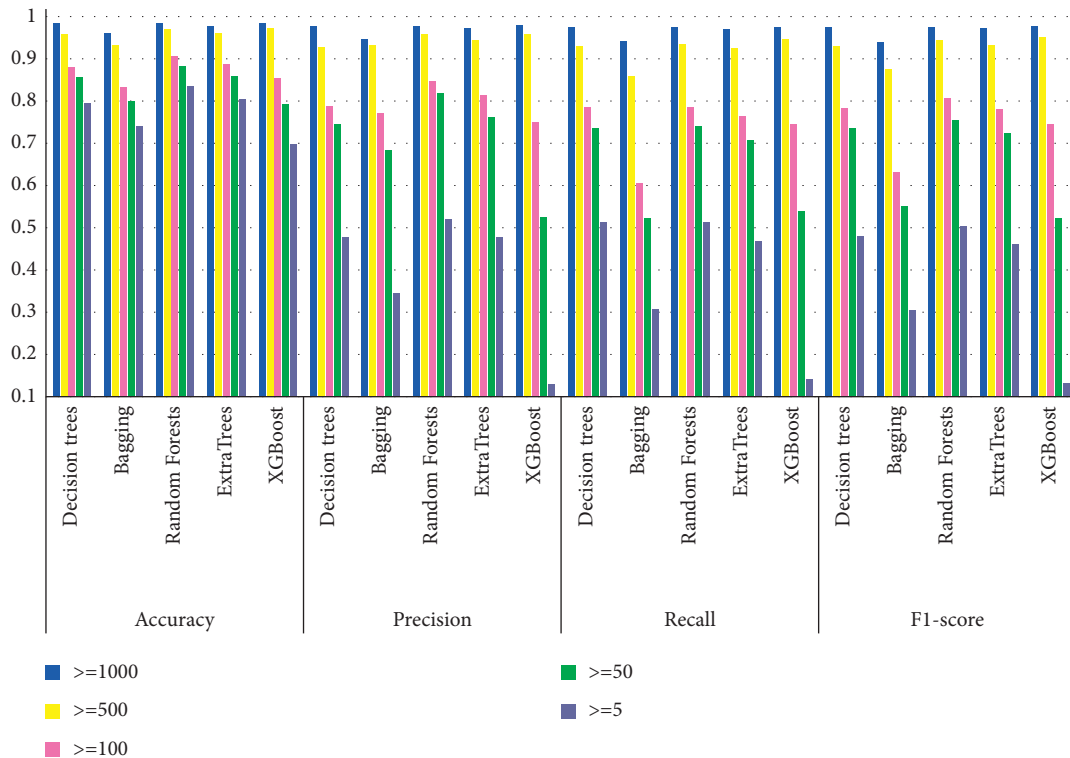


FIGURE 6: Overall evaluation and comparison of 5 models.

matrix is a diagonal matrix. Otherwise, nonzero numbers in the off-diagonal positions of the confusion matrix represent the number and result in incorrect predictions. For instance, the value 1 in row 2, column 9 indicates that a terrorist attack by the African National Congress (South Africa) was

incorrectly predicted as the Corsican National Liberation Front (FLNC).

From Figure 8, it can be observed that the confusion matrix was almost diagonally symmetric. The terrorist organizations with the most predicted errors were the National

TABLE 5: Comparison of 10-fold cross-validation and hold-out methods (terror attack frequency ≥ 500).

Metrics		Data split verification method			
		Hold-out method	10-fold cross-validation method		
			Mean	Max	Min
Accuracy	Decision trees	0.958647	0.956026	0.964387	0.949943
	Bagging	0.931989	0.933528	0.936764	0.927569
	Random forests	0.968216	0.965974	0.968744	0.962647
	ExtraTrees	0.959501	0.959406	0.962400	0.956011
	XGBoost	0.971634	0.967778	0.970828	0.963216
Precision	Decision trees	0.928242	0.929096	0.943624	0.918073
	Bagging	0.932152	0.927242	0.938100	0.911614
	Random forests	0.957727	0.955594	0.963753	0.947774
	ExtraTrees	0.942159	0.941508	0.945737	0.935197
	XGBoost	0.957246	0.952817	0.956800	0.943972
Recall	Decision trees	0.929554	0.931822	0.944081	0.923149
	Bagging	0.858106	0.862504	0.871233	0.854989
	Random forests	0.934140	0.934952	0.941422	0.929029
	ExtraTrees	0.926234	0.928944	0.934792	0.923533
	XGBoost	0.944904	0.942287	0.950476	0.933696
F1 score	Decision trees	0.928603	0.930063	0.943702	0.920590
	Bagging	0.875151	0.875903	0.886239	0.866793
	Random forests	0.942883	0.942658	0.949366	0.935658
	ExtraTrees	0.932798	0.933608	0.937390	0.927087
	XGBoost	0.950011	0.946542	0.953123	0.937474

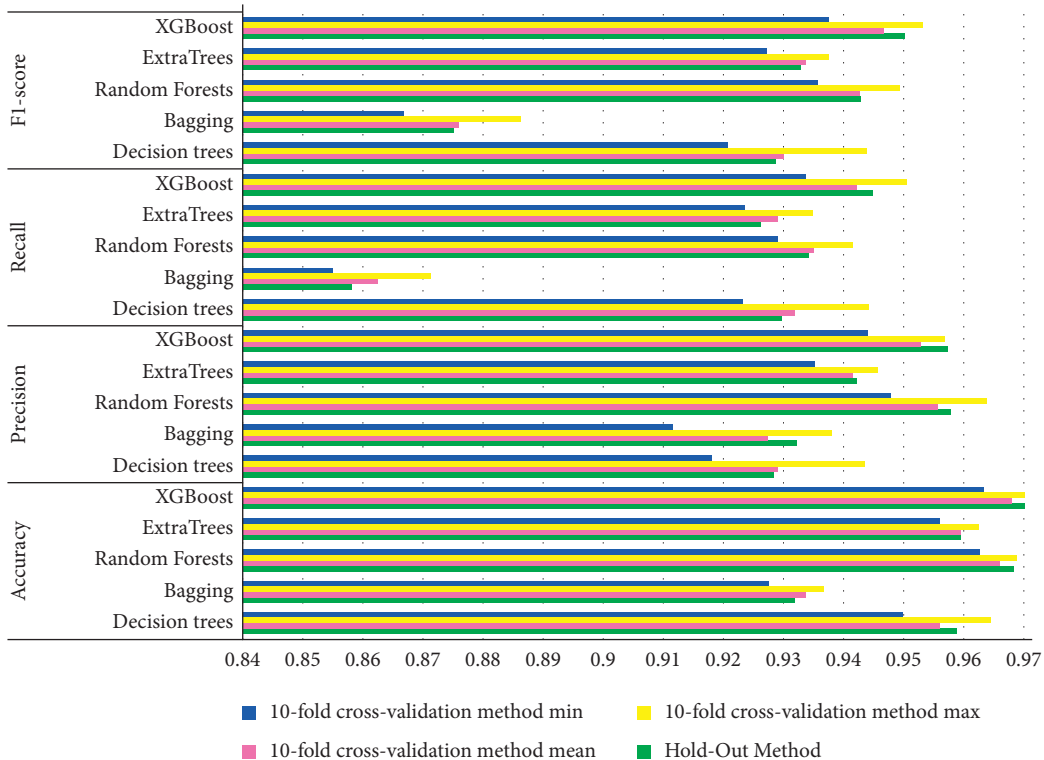


FIGURE 7: Comparison of model metrics using two data split verification methods.

Liberation Army of Colombia (ELN) in line 11 from the bottom and the Revolutionary Armed Forces of Colombia (FARC) in line 7 from the bottom. 27 attacks by the ELN were erroneously predicted as FARC, and 25 attacks of FARC were erroneously predicted as ELNs. These values are

close to diagonally symmetric. The reason is that these two terrorist organizations have similar features, so the prediction model can easily be confused with them. Similarly, the organizations FARC and M-19 (Movement of April 19) were also similar, with 9 and 14 prediction errors,

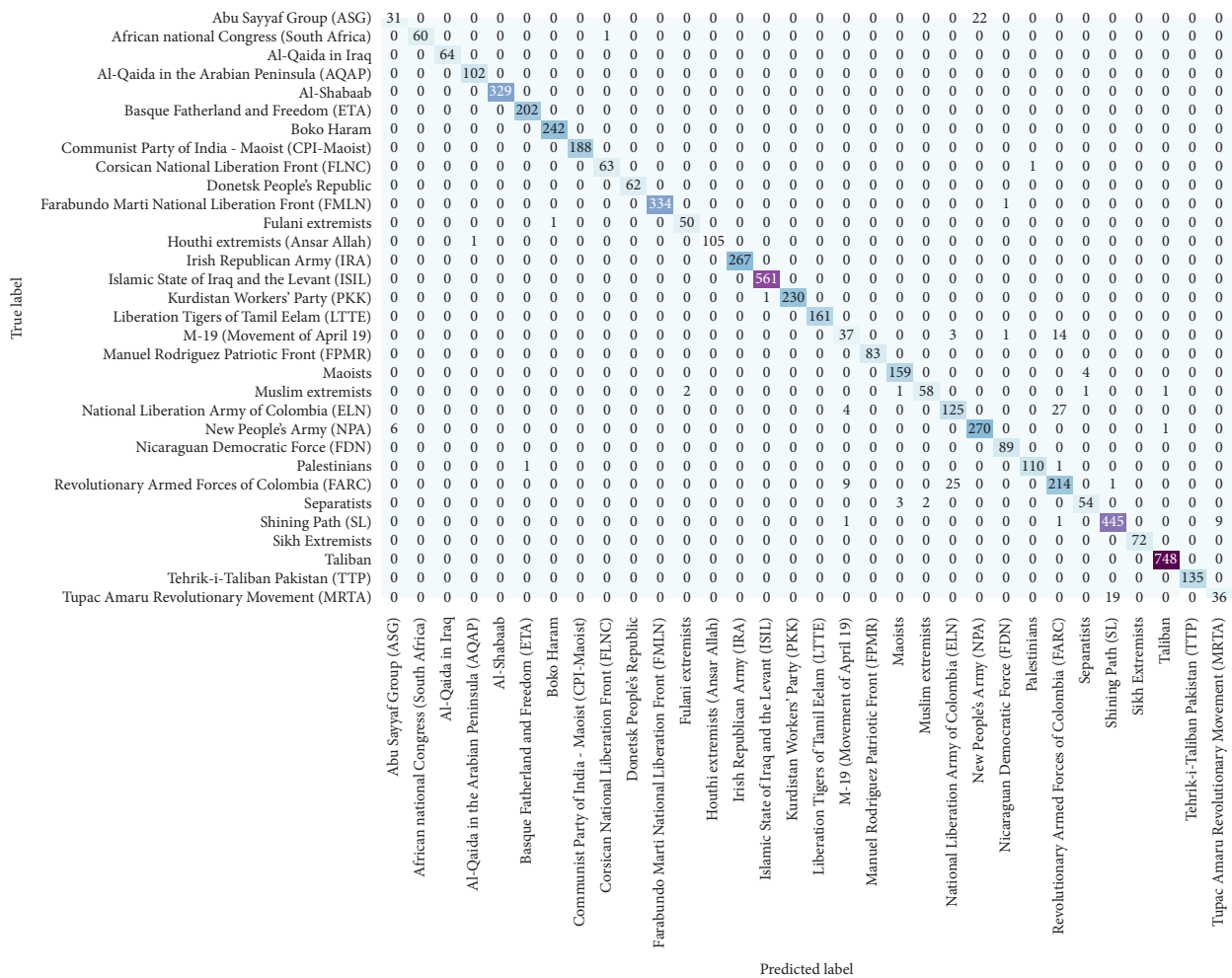


FIGURE 8: Confusion matrix of 32 terrorist organization prediction of the XGBoost model.

respectively. Therefore, it may be observed that FARC was more easily incorrectly predicted than other organizations, and its prediction precision and recall were correspondingly lower.

6. Conclusion

In this study, through a quantitative analysis of the data in the GTD, ensemble machine learning has been used to construct five multiclass classification models for the prediction of terrorist organizations that perpetrated terrorist attacks.

First, according to the frequency of terrorist organization attacks, the terrorist organizations were analyzed, and the characteristics and trends of 32 terrorist organizations with more than 500 terrorist attacks were described in detail. Then, for the prediction of terrorist organizations in terrorist attacks, 36 feature attributes were selected based on the feature selection strategy, and five classifiers, including decision tree, bagging, random forest, extra tree, and XGBoost, were constructed to predict terrorist organizations. The performance and stability of the five models were evaluated using hold-out and 10-fold cross-validation

methods, respectively. Our models predicted 32 terrorist organizations for high-frequency activities in terrorist attacks. Finally, the experimental results showed that the five models achieved good performance and stability. XGBoost and the random forest classifier achieved the best prediction accuracies of 97.15% and 97.03%, respectively. We further visualized and analyzed the prediction results of the XGBoost model using the confusion matrix. Moreover, the method can be extended to the prediction of a broader range of terrorist organizations. Considering the number of terrorist organization classifications based on the frequency of attacks, the classification prediction accuracy of the random forest algorithms was consistently excellent. When the number of terrorist organizations was small (e.g., dozens), XGBoost exhibited the best prediction accuracy, and the performance of random forest was close to that of XGBoost.

The prediction model presented herein can macroscopically predict the terrorist organizations of global terrorist attacks, excavate the relevant factors of terrorist attacks, and provide decision support for the prevention and control of antiterrorism organizations and related countries. With further improvement in the performance and accuracy of machine learning algorithms, we believe that these

technologies can help security departments find better algorithmic models and appropriate datasets to improve the accuracy of predictions related to terrorist attacks. However, considering the local sparsity of terrorist attacks and their versatility in planning and execution, even with the continuous progress of machine learning, research on large-scale monitoring and prediction algorithms is nonetheless expected to be challenging.

Data Availability

The dataset used in this work is derived from the data on terrorist attacks from 1970 to 2017 in the Global Terrorism Database (GTD) [2]. The GTD dataset is built by the United States Anti-Terrorism Research Consortium (START) and the University of Maryland and is publicly available at <https://www.start.umd.edu/gtd/>.

Conflicts of Interest

The author declares no conflicts of interest.

Acknowledgments

This work was supported in part by the Philosophy and Social Science Planning Project of Shanghai, under Grant no. 2019BGL028.

References

- [1] J. M. Poland, *Understanding Terrorism: Groups, Strategies, and Responses*, Prentice-Hall, Englewood Cliffs, 1988.
- [2] G. LaFree and L. Dugan, "Introducing the global terrorism database," *Terrorism and Political Violence*, vol. 19, no. 2, pp. 181–204, 2007.
- [3] F. Ding, Q. Ge, D. Jiang, J. Fu, and M. Hao, "Understanding the dynamics of terrorism events with multiple-discipline datasets and machine learning approach," *PloS One*, vol. 12, no. 6, Article ID e0179057, 2017.
- [4] Y.-L. Chuang, N. Ben-Asher, and M. R. D'Orsogna, "Local alliances and rivalries shape near-repeat terror activity of al-Qaeda, ISIS, and insurgents," *Proceedings of the National Academy of Sciences*, vol. 116, no. 42, pp. 20898–20903, 2019.
- [5] V. B. Petroff, J. H. Bond, D. H. Bond, and D. H. Bond, "Using hidden Markov models to predict terror before it hits (again)," in *Handbook of Computational Approaches to Counterterrorism*, pp. 163–180, Springer, New York, NY, USA, 2013.
- [6] F. Gohar, W. H. Butt, and U. Qamar, "Terrorist group prediction using data classification," in *Proceedings of the International Conference on Artificial Intelligence and Pattern Recognition*, pp. 199–208, Kuala Lumpur, Malaysia, 2014.
- [7] G. M. Tolan, O. S. Soliman, and O. S. Soliman, "An experimental study of classification algorithms for terrorism prediction," *International Journal of Knowledge Engineering-IACSIT*, vol. 1, no. 2, pp. 107–112, 2015.
- [8] X. Meng, L. Nie, and J. Song, "Big data-based prediction of terrorist attacks," *Computers & Electrical Engineering*, vol. 77, pp. 120–127, 2019.
- [9] B. Bu, Z. Pi, and L. Wang, "Support vector machine for classification of terrorist attacks based on intelligent tuned harmony search," *Ekoloji*, vol. 28, no. 107, pp. 153–164, 2019.
- [10] Z. Li, D. Sun, B. Li, Z. Li, and A. Li, "Terrorist group behavior prediction by wavelet transform-based pattern recognition," *Discrete Dynamics in Nature and Society*, vol. 2018, Article ID 5676712, 2018.
- [11] X. Hu, F. Lai, G. Chen, R. Zou, and Q. Feng, "Quantitative research on global terrorist attacks and terrorist attack classification," *Sustainability*, vol. 11, no. 5, p. 1487, 2019.
- [12] G. M. Campedelli, M. Bartulovic, and K. M. Carley, "Learning future terrorist targets through temporal meta-graphs," *Scientific Reports*, vol. 11, no. 1, pp. 8533–8615, 2021.
- [13] G. M. Campedelli, *On Meta-Networks, Deep Learning, Time and Jihadism*, Università Cattolica del Sacro Cuore, XXXII ciclo, a.a. 2018/19, Milano <http://hdl.handle.net/10280/70552>.
- [14] G. M. Campedelli, I. Cruickshank, and K. M. Carley, "A complex networks approach to find latent clusters of terrorist groups," *Applied Network Science*, vol. 4, no. 1, pp. 1–22, 2019.
- [15] B. A. Desmarais and S. J. Cranmer, "Forecasting the locational dynamics of transnational terrorism: a network analytic approach," *Security Informatics*, vol. 2, no. 1, pp. 1–12, 2013.
- [16] W. Guo, K. Gleditsch, and A. Wilson, "Retool AI to forecast and limit wars," *Nature*, vol. 562, no. 7727, pp. 331–333, 2018.
- [17] S. B. Kotsiantis, I. Zaharakis, and P. Pintelas, "Supervised machine learning: a review of classification techniques," *Emerging artificial intelligence applications in computer engineering*, vol. 160, pp. 3–24, 2007.
- [18] S. R. Safavian and D. Landgrebe, "A survey of decision tree classifier methodology," *IEEE transactions on systems, man, and cybernetics*, vol. 21, no. 3, pp. 660–674, 1991.
- [19] P. E. Utgoff, N. C. Berkman, and J. A. Clouse, "Decision tree induction based on efficient tree restructuring," *Machine Learning*, vol. 29, no. 1, pp. 5–44, 1997.
- [20] R. J. Lewis, "An introduction to classification and regression tree (CART) analysis," in *Annual Meeting of the Society for Academic Emergency Medicine in San Francisco*, vol. 14, Addison-Wesley Educational, California, USA, 2000.
- [21] L. Rutkowski, M. Jaworski, L. Pietruczuk, and P. Duda, "The CART decision tree for mining data streams," *Information Sciences*, vol. 266, pp. 1–15, 2014.
- [22] T. G. Dietterich, "Ensemble learning," *The handbook of brain theory and neural networks*, vol. 2, pp. 110–125, 2002.
- [23] L. Breiman, "Bagging predictors," *Machine Learning*, vol. 24, no. 2, pp. 123–140, 1996.
- [24] M. Galar, A. Fernandez, E. Barrenechea, and H. Sola, "A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 4, pp. 463–484, 2011.
- [25] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.
- [26] P. Geurts, D. Ernst, and L. Wehenkel, "Extremely randomized trees," *Machine Learning*, vol. 63, no. 1, pp. 3–42, 2006.
- [27] L. Wehenkel, D. Ernst, and P. Geurts, "Ensembles of extremely randomized trees and some generic applications," in *Proceedings of Robust Methods for Power System State Estimation and Load Forecasting*, Paris, 2006.
- [28] T. Chen and C. Guestrin, "Xgboost: a scalable tree boosting system," in *Proceedings of the 22nd ACM sigkdd international conference on knowledge discovery and data mining*, pp. 785–794, San Francisco, CA, USA, August 2016.
- [29] T. Chen, T. He, and M. Benesty, *Xgboost: Extreme Gradient Boosting*, pp. 1–4, 2015, R package version 0.4-2.
- [30] Y. Kong and M. Jing, "Research of the classification method based on confusion matrixes and ensemble learning," *Computer Engineering & Science*, vol. 34, no. 6, p. 111, 2012.

- [31] F. Pedregosa, G. Varoquaux, A. Gramfort, and V. Michel, "Scikit-learn: machine learning in python," *The Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.
- [32] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *Journal of Machine Learning Research*, vol. 13, no. Feb, pp. 281–305, 2012.

Research Article

Grade Prediction in Blended Learning Using Multisource Data

Ling-qing Chen ¹, Mei-ting Wu,¹ Li-fang Pan ^{2,3} and Ru-bin Zheng¹

¹School of Computer Engineering, Jimei University, Xiamen 361021, China

²School of Science, Jimei University, Xiamen 361021, China

³Digital Fujian Big Data Modeling and Intelligent Computing Institute, Jimei University, Xiamen 361021, China

Correspondence should be addressed to Li-fang Pan; xiaopan@jmu.edu.cn

Received 10 June 2021; Accepted 30 August 2021; Published 14 September 2021

Academic Editor: KunHong Liu

Copyright © 2021 Ling-qing Chen et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Today, blended learning is widely carried out in many colleges. Different online learning platforms have accumulated a large number of fine granularity records of students' learning behavior, which provides us with an excellent opportunity to analyze students' learning behavior. In this paper, based on the behavior log data in four consecutive years of blended learning in a college's programming course, we propose a novel multiclassification frame to predict students' learning outcomes. First, the data obtained from diverse platforms, i.e., MOOC, Cnblogs, Programming Teaching Assistant (PTA) system, and Rain Classroom, are integrated and preprocessed. Second, a novel error-correcting output codes (ECOC) multiclassification framework, based on genetic algorithm (GA) and ternary bitwise calculator, is designed to effectively predict the grade levels of students by optimizing the code-matrix, feature subset, and binary classifiers of ECOC. Experimental results show that the proposed algorithm in this paper significantly outperforms other alternatives in predicting students' grades. In addition, the performance of the algorithm can be further improved by adding the grades of prerequisite courses.

1. Introduction

In recent years, online education platforms have increased with the rapid development of Internet technology; and the mixed teaching mode combining offline face-to-face teaching and online learning has also emerged. Mixed teaching, aided with computers, mobile phones, and other electronic equipment, guides students to carry out personalized and adaptive learning according to teachers' preset learning goals and driving tasks. Under the mixed teaching mode, students self-regulate the learning process, and teachers supervise students' learning progress as well as efficiency in real time. Integrating online evaluation with offline evaluation, the students' learning autonomy increases while learning efficiency is guaranteed. At the same time, combining with the analysis of students' online learning behavior trajectory and offline classroom performance, we can infer the students' phased learning situation to adjust the teaching pace in time and further improve the teaching quality. Therefore, it is worth studying how to transform students' learning data such as their online learning behavior

track and offline classroom performance into their phased learning situation and take corresponding measures to adjust and guide their personalized learning to achieve better teaching effect.

The structure of this paper is organized as follows. Section 2 reviews related literature. Section 3 describes the setting of the curriculum and data from the platforms. Section 4 provides a brief introduction to error-correcting output codes (ECOC) and genetic algorithms (GA) and then further illustrates the design and implementation of the TBCGA-ECOC (Ternary Bitwise Calculator based GA for ECOC) algorithm. Section 5 describes the experimental parameters and analyzes the experimental results of various algorithms. Section 6 summarizes the work.

2. Literature Review

Under the upsurge of mixed teaching, many scholars have conducted researches on related fields. Yousef and Sumner conducted detailed analysis and discussion on more than 200 articles on MOOC. They proposed possible future

research directions according to the main achievements and MOOC research direction in recent years [1]. For emerging ways of education big data, Ang et al. introduced and discussed six aspects including overview and classification of education big data, different data analysis methods, social challenges and technical challenges, technical problems, systematic data sources, data mining, and database [2]. To explore the underlying mechanisms behind the features, Xie conducted a survival analysis of the viewing duration of large-scale open online courses, in which two stochastic differential equations were used to describe the growth of two classes. The research helps in understanding the role of memory in the complexity of learning behavior [3]. Based on SPOC and flipped classes, Xu et al. also launched research on online platforms. They created experiments on the computer network mixed classes, stating that students' online behavior data can be used to predict their results and the results are more stable and reliable as classes advance [4]. Guided by Moore's trading distance, Hew et al. defined MOOC success as student satisfaction with the course [5]. They proposed a unique methodology framework (supervised machine learning and sentiment analysis) to examine user review generated datasets and identify specific learners level and course level factors. It can predict MOOC learners' satisfaction and estimate their relative impact. The research extended the theoretical understanding of the factors affecting learner MOOC satisfaction and presented specific opinions for the design of MOOC. To adjust the video curriculum and arrangement of MOOC better, Hu et al. put forward the analysis method of students' video viewing behavior based on the Spark platform and verified it with the data of the cauX platform. The experimental results show that the method can analyze the characteristics of the viewing behavior quickly and accurately [6]. Yu et al. have noted that SPL-based methods can be used to establish a practical MOOC learning analysis framework, which increased the complexity and reusability of learning analysis [7]. Zhang et al. constructed a SPOC-based flipped classroom teaching model based on the OBE concept and verified its effectiveness [8]. In order to build a better MOOC teaching platform and provide theoretical guidance and policy advice for college managers, Wang et al. combined the technology acceptance model (TAM) and plan behavior theory (TPB) to construct a theoretical model to analyze the mechanism behind MOOC learning performance [9]. It forms a scientific basis for the MOOC teaching setting and provides theoretical guidance and policy advice for college managers. Meanwhile Gardner and Brooks classified MOOC studies by predictors, predictions, and underlying theoretical models [10]. Then they critically investigated the work of each category and provided data on raw data sources, feature engineering, statistical models, evaluation methods, prediction architectures, and other aspects of these experiments, revealing several critical approach gaps and exploring different features and modeling technology spaces. The results make valuable contributions to construct accuracy, operability, and student success models for theoretical construction. A document clustering model based on weighted word embeddings was developed by Onan and Toçoğlu to

identify the post problem on MOOC forums [11]. Additionally, Toçoğlu and Onan designed a long short-term memory network (LSTM) which can classify around 70,000 MOOC reviews more accurately [12]. They also investigated the sentiment analysis of Twitter data and student evaluation data [13–15].

In addition to the research on MOOC data and teaching mode, many scholars focused on students' behavior models, researched student performance prediction, and explored factors affecting student study. Xu et al. developed a two-layer structure model consisting of multiple underlying predictors and cascading integrated predictors [16]. They also proposed a data-driven approach based on latent factor models and probability matrix decomposition to discover the curriculum relevance to construct more efficient underlying predictors. It is demonstrated that the proposed method performs the benchmark method. In addition, Ulloa-Cazarez et al. proposed genetic planning (GP) algorithm to predict whether students can pass the final exam for early warning of their learning status [17]. Xu and Yang designed classification algorithms based on support vector machine (SVM) to summarize students' learning motivation through their MOOC activity log to predict whether they can obtain a certificate [18]. Similar work was being done by Meier et al. [19]. They predicted possible student performance in subsequent learning through historical teaching data from the course and took instructional intervention when necessary. Qiu et al. proposed the latent dynamic factor graph (LadFG) to predict student homework completion and whether students can get a certificate successfully [20]. Based on the MOOC data established by Peking University on Coursera, Zhang et al. for the first time classified and analyzed student behavior characteristics in Chinese MOOC and predicted learning results with three classification models: linear discriminant analysis (LDA), logical regression (LR), and linear nuclear support vector machine (l-SVM) [21]. Yu et al. identified seven cognitive participation models based on students' video clickstream logs. Also, they designed classification algorithms based on K-nearest neighbors (KNN), SVM, and artificial neural networks (ANN) to predict whether students can pass the course exam [22]. Mubarak et al. employed a dual analysis method combining visual and predictive analysis to visualize the data and model the behavior with an RNN-LSTM model, which improved the prediction accuracy of learning performance [23]. In addition, Wen et al. proposed a new simple feature matrix to maintain local correlation information with learning behavior and a novel Convolutional Neural Network (CNN) model to improve the completion rate of MOOC [24]. Hussain et al. analyzed data recorded in digital electronic Education and Design Suite (Contract) using five machine learning algorithms: artificial neural network (ANN), support vector machine (SVM), logical regression, naive Bayesian classifier, and decision tree [25]. ViSeq, an interactive visual analysis system available to visualize the learning order of different groups of learners proposed by Chen et al., helped users explore the learning sequence of MOOC from multiple granularity levels [26]. It includes four-link maps: projection views of

learner-identifying groups, pattern views showing the overall sequence patterns within the selected group, sequence views describing transformations between ongoing events, and individual views with an extended sequence chain. Cobo and Ruiz-Garcia presented the edX-LIS Learning Intervention System, which provides intervention strategies to improve learning performance, positively affecting learning motivation, persistence, and participation [27]. García-Molina et al. presented an automatic scoring algorithm based on the data dimension and correlation of learners' contributions [28]. It provides teachers with visual and digital representation for each learner, offering opinions for adjusting the expected behavior in the MOOC forum. In response to the rapid change in knowledge in the IT domain, Chen et al. proposed a framework for in-based expression through decision tree expression to evaluate the relationship between the curriculum and employment [29]. They designed the algorithm recommended by the course group based on the decision to provide the basis for adjusting the course configuration. The results indicate that the designed algorithms and recommended curriculum groups have a significant role in promoting student employment.

In essence, these studies are the integration of multiple classification problems. They analyzed the collected data (behavioral trajectory data) with different models or algorithm frameworks and finally offered predicting outcomes. Many studies have also proved the feasibility and effectiveness of relevant algorithms and models in result prediction, which provided a solid theoretical basis for the application and improvement of algorithms. This paper mainly collects students' learning behavior track data and divides their grades into excellent, qualified, and unqualified grades. We add feature selection based on GA and based classifier selection to the ECOC algorithm. We also optimize the ECOC coding matrix, feature subset, and choice of binary classifiers, taking accuracy as the evaluation index. By screening the learning behavior characteristics of raw data in the data preprocessing stage, the multiclassified accuracy can be improved. Next, we get the learning behavior features that influenced student performance to analyze through the fine-grained results of feature selection and correlation analysis. Finally, according to the character selection and correlation analysis of fine-grained results, the teaching methods can be improved and the teaching rhythm can be adjusted, which will be helpful in guiding the students' personalized learning and improve their learning performance.

3. Data Description

3.1. Course Settings. The datasets for this analysis were collected from a blended course of *Java Language Programming* in a college for four consecutive years. To make students understand the basic knowledge of Java and master the Java programming technology, this course integrates the characteristics of Java and introduces the programming foundation, object-oriented programming, GUI programming, and so forth; and it is scheduled for 68 class hours. Unlike traditional classes, it adopts the current popular online learning platforms such as rain classroom and

MOOC for auxiliary teaching. In addition to participating in conventional offline classes, students also need to use online platforms to preview, review, and complete tests or do some other online learning activities and take the final exam offline in the end.

3.2. Data Description. The course data are the learning track and data records collected from a blended course of *Java Language Programming* in a college for four consecutive years (the four different grades are represented by grade-1, grade-2, grade-3, and grade-4). The data source consists of four popular online education platforms: MOOC Platform, Rain Classroom, Programming Teaching Assistant (PTA), and Cnblogs.

Four grades (grade-1, grade-2, grade-3, and grade-4) have 94, 127, 130, and 122 people participating in the courses, respectively. Auxiliary teaching platforms for each grade are depicted in Figure 1.

3.2.1. Data of MOOC. The MOOC Platform data mainly includes the viewing of 102 teaching videos, the scores of 6 mutual-evaluation homework and 8 chapter tests, the cumulative number of discussions in the current semester, the learning times of each chapter, and the online test results as well as submission time. The data field is summarized in Table 1.

3.2.2. Data of Rain Classroom. Rain Classroom is an intelligent online teaching platform covering a variety of preclass-classroom-after-class teaching scenes. The collected data mainly include preclass preview situations and classroom performance. Tables 2 and 3, respectively, list the data composition and field description.

3.2.3. Data of Cnblogs. Cnblogs is a knowledge-sharing community for developers where students can consolidate knowledge and regularly make a summary by writing blogs. At the same time, teachers and teaching assistants can review students' blogs to understand their phased learning results and current situations. The main components of the scores include chapter knowledge summary, PTA exercise score situation, PTA exercise algorithm analysis, code reading, and learning feelings. Figure 2 portrays the composition of the Cnblogs' score.

3.2.4. Data Integration. In addition to the three categories of data sources, PTA online tests were added to integrating data. When processing the data, we need to align the data from different sources with the student ID as the standard and then combine the data to get four complete datasets. At the same time, the student's grades are divided into three grades, excellent, qualified, and unqualified, for the subsequent grades and grade prediction.

After integration, the pretreatment of the data (filtering, feature conversion) was conducted to obtain the final datasets. Table 4 lists the four datasets.

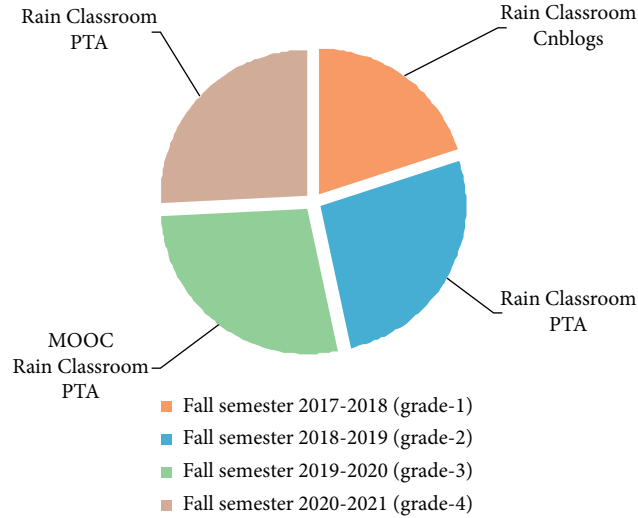


FIGURE 1: The source and composition of the data.

TABLE 1: The data of MOOC.

Feature types	Number	Data description	Examples
Teaching video viewing situation	102	The completion of watching each teaching video is presented in percentage	100%, 55%
Student information	7	School number, name, and other identifying information	
Discussion	4	The number of postdiscussions and reply discussions are presented by the number of replies	0, 5
Job scores	6	Job score, a total score of 5	0, 4.5, 5
Job completion status	6	Complete or not	Yes/no
Job mutual-evaluation status	6	Rate each other or not	Yes/no
Scores on tests	8	Online test scores, out of 100	0, 90.0, 100
Test completion time	8	Test completion time in YYYY-MM-DD, hh:mm:ss	2019-12-28, 00:24:09
Watching time duration	1	Student cumulative time for the video, in minutes	721 minutes
Accumulated learning times	1	Number of times students open a learning platform to learn	324
Progress of task point completion	4	Progress of video viewing and completion of the test, in scores	120/120, 1/8
Certificate issuance	1	The MOOC Platform can apply for certification after completing the specified indicators	Yes/no
The score of online platforms	2	Regular results and total results on the teaching platform, out of 100	90, 100
Five levels	1	Convert scores to grade system	A, B, C, D, E

TABLE 2: Data composition of the Rain Classroom.

Semester/grade	Preclass preview	Classroom performance
Fall semester 2017-2018 (grade-1)	25	9
Fall semester 2018-2019 (grade-2)	27	16
Fall semester 2019-2020 (grade-3)	26	15
Fall semester 2020-2021 (grade-4)	26	11

4. The ECOC Framework Based on GA and TBC

The multiple classification problem in the field of machine learning is often divided into multiple double classification problems. Finally, the multiple classifications are accomplished by integrating all the results. This paper focuses on ECOC classification-based algorithm by splitting multiple classification problems and combining multiple two classifiers.

4.1. Introduction to ECOC. ECOC algorithm, originated in the field of digital communication, is an ensemble learning method framework for multiclass classification. However, it is widely used in face recognition, traffic signal recognition, microarray data analysis, and other fields nowadays. In the multiclassification problem, by designing proper coding for each category, the ECOC ensemble system can achieve the error correction function by using the classifier to classify the samples into the correct category [30].

TABLE 3: Description of the rain classroom data.

Type	Field	Data description	Example
Essential information	Name	Student information	San Zhang
	Student ID	Student information	20XX21123XXX
	Total points	Total results in rain classroom (full marks are different)	100
	Sign-in times	Total number of classroom check-ins	11
	Access rate	Class check-in rate (percentage)	100%
Preview before class	Number of announcements read	Number of readings of published announcements	15
	Total number of pages viewed	Number of viewing pages of the pushed courseware	10
	Total viewing time length	Watching time of the pushed courseware XX h, XX m, XX s	00 h, 00 m, 00 s
	Finish time	Time to complete the courseware preview YYYY:MM:DD HH:MM:SS or completion status	Time/no preview/not completely finish
	Answer score	The score of the pushed courseware exercises	10
Classroom performance	Sign-in mode	The score of the pushed courseware exercises	Not signed in/scan the code to sign in
	Barrage releases	Number of barrage releases	0
	Accumulated score	Integration of classroom performance scores	10

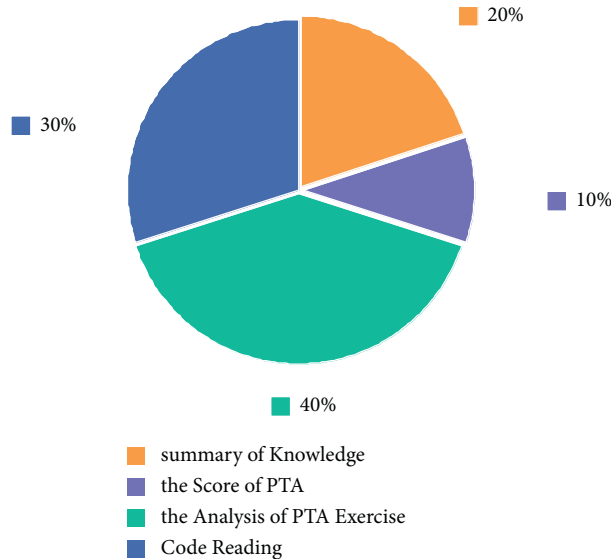


FIGURE 2: Composition of the Cnblogs' score.

The ECOC algorithm mainly includes three basic steps [31]: encoding, training, and decoding. The process of constructing a coding matrix is called coding. The coding method is roughly divided into two categories: data-dependent and data-independent. The former refers to the model or distribution of the datasets during the coding process, typically including ECOC-ONE, DECOC (Discriminant ECOOC), and Data-Driven ECOC. The data-independent coding process only considers the number of categories. It does not analyze the model and distribution of the data, typically including OVO (One Versus One) [32], OVR (One Versus Rest), OVA (One Versus All), DR (Dense Random) [33], SR (Sparse Random) [34], and other coding methods [35]. The coding matrix consists of “+1,” “-1,” or “0.” Each coding matrix represents a category, and each column represents a binary classifier. Two classifiers train only the non-0 encoding corresponding category data and take “+1,” encoding corresponding category

training data as positive classes and “-1” corresponding category data as negative classes. Figure 3 portrays a classical one-to-many encoding (One vs All, OVA) for the encoding matrix. The ECOC algorithm is essentially an integration framework that improves model performance by integrating different classifiers and achieves classifier complementarity [36, 37].

The process opposite to coding is called decoding. The key to decoding is to determine the final category of the unknown sample based on the decision results of each base classifier. Decoding mainly consists of three types: decoding strategies based on output coding and target coding distance such as Hamming distance decoding, probability-based decoding such as Bayesian criterion, and mode space-based decoding strategy.

4.2. Feature Selection Method. Feature selection is a primary data processing method in machine learning [38, 39]. It improves the accuracy of prediction models and constructs

TABLE 4: Description of integration data summary.

Semester/grade	Data dimensions	Number of pieces of data
Fall semester 2017-2018 (grade 1)	119	82
Fall semester 2018-2019 (grade 2)	130	117
Fall semester 2019-2020 (grade 3)	130	115
Fall semester 2020-2021 (grade 4)	131	119

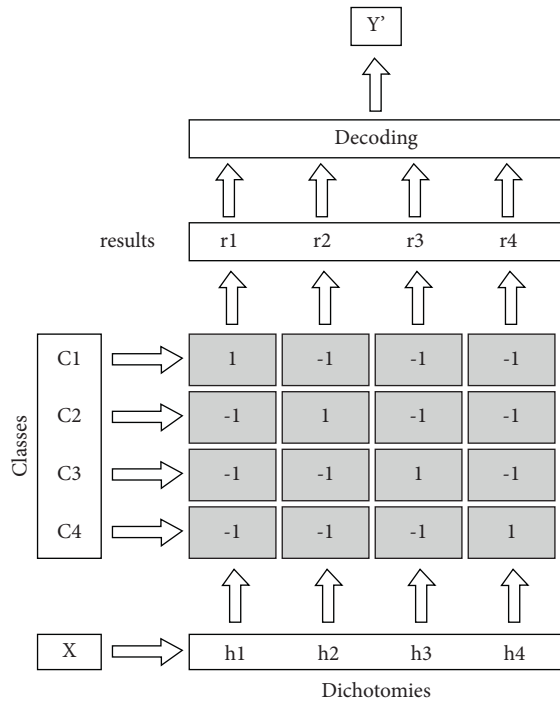


FIGURE 3: Schematic diagram of the OVA encoding matrix.

faster and less consuming prediction models and gives the model for better understanding and interpretation by selecting some features from all features for training models, enhancing the generalization ability of the model, highlighting the essence of problems, and simplifying the mathematical model.

The origins of GA date back to the early 1960s. GA searches for the optimal solution by simulating a natural evolutionary process. The problem-solving process is converted to the crossover and variation of chromosome genes in biological evolution through computer simulation operation. The algorithm has three basic operators: selection, crossover, and variation. Compared with traditional algorithms, GA is based on biological evolution with good convergence, less computational time, and high robustness in computational accuracy requirements. Meanwhile, the algorithm has scalability and is easy to combine with other algorithms [40].

4.3. The ECOC Framework Based on GA and TBC. The classical ECOC algorithm takes all the features as the input variables and uses a collection of base classifiers to train and predict the model. In this paper, based on ECOC and GA for performance grade prediction, we design a multi-classification algorithm framework. The algorithm performs

the optimization process to exchange their information by the vector operator computing code words between a pair of encoded columns; and it employs a collection of heterogeneous classifiers via GA for better solutions.

4.3.1. Gene. Gene is a feature in the solution of the problem. In this paper, a gene can guide encoding columns to computation to generate a new encoding column. The gene design mainly includes the index of the two selected matrix columns, the feature selection sequence, and two columns for calculation. Gene structure is illustrated in Figure 4.

4.3.2. Individual. An individual is consisted of a series of genes. It is a unit that needs to evolve or a solution to the original problem. In this algorithm, a new ECOC coding matrix can finally be generated through an individual by integrating new generated coding columns of genes; and the number of two classifiers determines the number of genes in the individual. The design of the individual is shown in Figure 5.

4.3.3. Operator. The algorithm has five types of operators. It is mainly used to operate the selected characteristic matrix column according to the chosen operator and generate a new characteristic matrix.

4.3.4. Legality Check. The ECOC encoding matrix has its unique legitimacy constraints. Illegal cases mainly include duplicate rows, reverse columns, and rows with only 0 symbols. For column levels, it means that the column contains only 0 or contains only +1 or -1 or that +1 or -1 is missing.

It is necessary to conduct legality testing before performing the evaluation operation, correct illegal operations, and discard some illegal columns.

4.3.5. Crossover. Crossover and mutation are the basic steps in genetic algorithms. The primary purpose of variation is to increase the diversity of the population, thus avoiding local optimality due to gene monopoly after superior individuals. The algorithm requires the exchange of genes or parts between individuals with a certain crossover probability. The process of gene crossing is shown in Figures 6 and 7: select parent individuals to cross first and then exchange the same type of bits in two selected gene individuals.

4.3.6. Mutation. Mutations are used to maintain and introduce diversity in genetic groups; and the mutation probability determines the number of genes in the

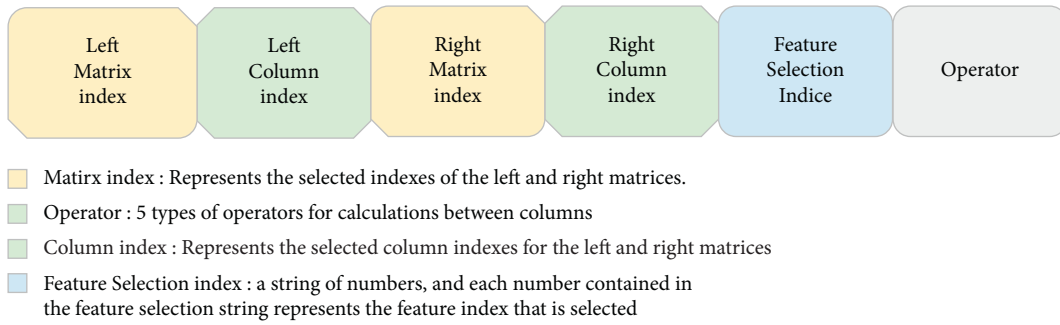


FIGURE 4: Gene structure.

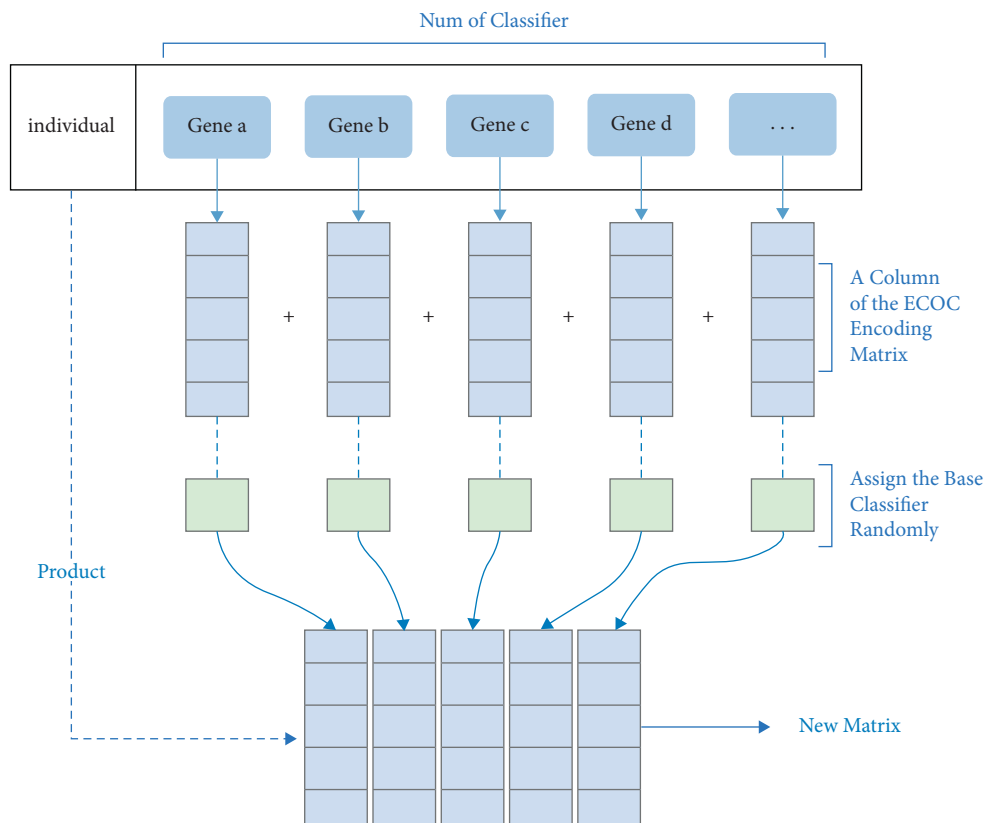


FIGURE 5: The representation of individual.

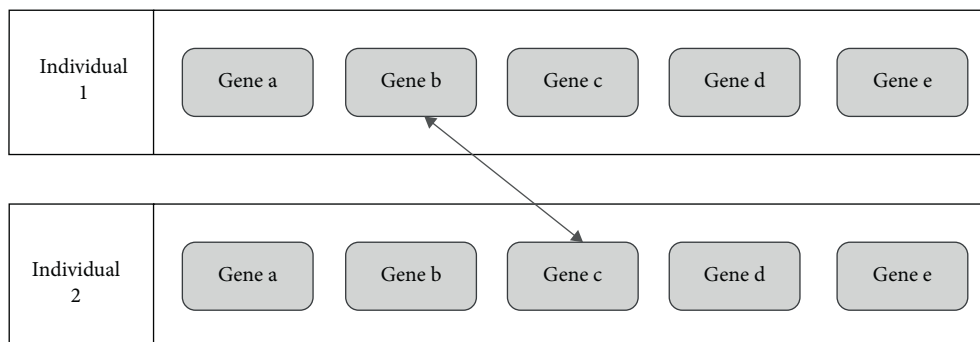


FIGURE 6: Select parent individuals to cross.

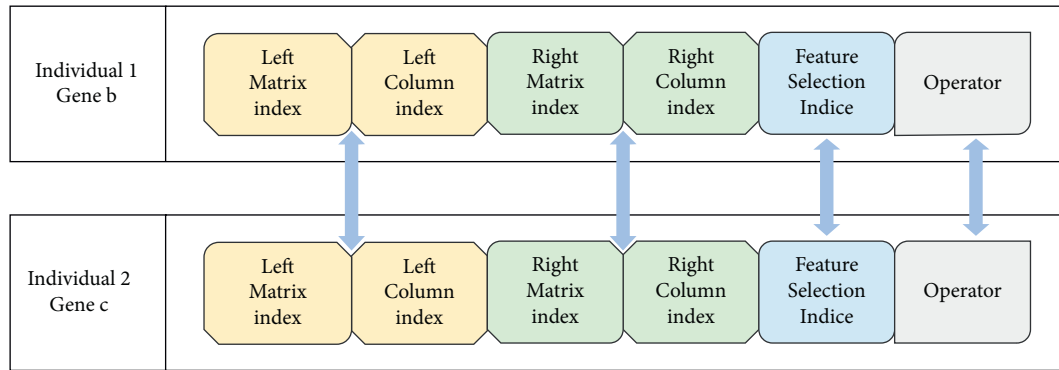


FIGURE 7: Exchange the same type of bits in two selected gene individuals.

individuals selected for the mutation. Each gene has a low mutant mutation probability. For the chosen gene, each of them will change randomly with a mutation probability.

If the matrix index is selected to change, the original number will be changed within all the indexes of the encoding matrix pool. If the column index in the selection matrix is changed, do so within the number of columns. If the feature selection sequence is selected, some of the bits are selected in the digital sequence for mutation representing the feature selection index mutation of the classification model. If the operator is selected, change it in the five operator index ranges.

Figure 8 describes the example of mutation.

4.3.7. Elite Retention. The algorithm adopts an elite retention strategy to ensure that the classification effect rises during the iteration process and to avoid the optimal individual being destroyed by crossover and mutation. Suppose that the individuals with the worst adaptability in the new generation population are worse than the optimal individuals in the previous generation population. In that case, the worst individuals in the new generation will be replaced with the optimal individuals of the last generation to ensure the upward trend of evolution. In the evolution of genetic algorithms, the optimal individuals appearing to date are not lost or destroyed by the selection, crossover, and variation operations. Elite retention strategies play a significant role in improving the global convergence capabilities of standard genetic algorithms.

4.3.8. Optimization Method of the Binary Classifiers. The algorithm uses a hybrid classifier that places multiple classifiers into the chromosome and jointly iterates with the population to select the optimal base classifier corresponding to each column of each matrix. Each column is trained with the most optimum base classifier and combines the corresponding matrix column into a new matrix to obtain an optimal matrix.

4.3.9. Flowchart of the Proposed Algorithm

- (1) Relevant parameter setting: initialize all kinds of parameters, such as population iteration number,

crossover probability, mutation probability, the number of individuals in each generation of population, and the number of the initial coding matrices in the pool of coding matrix.

- (2) Population: A population is a collection of several individuals. It is the unit of iteration in the algorithm; and the evolution of the final result is conducted in population units.
- (3) Population initialization: The population contains several ECOC encoding matrix individuals. When generating a primary population, generate illegitimate ECOC coding matrices as primary individuals according to the constraint law of the ECOC matrix randomly.
- (4) Assess the new individual: The model takes accuracy as the index to calculate the classification effect of the model and performs the “survival of the fittest” operation.
- (5) Termination condition: Terminate the algorithm when the number of iterations reaches the preset value. The solution is now the optimal solution.

The algorithm is verified by the 5-fold cross and based on ECOC for model training and prediction. The average accuracy is used as the fitness function to measure the performance improvement of the multiclassification problem. In the iterative process of the algorithm, we select the more adaptable feature matrix and base classifier sequence for subsequent iterations. The final individual is the optimal individual. The algorithm’s flowchart is shown in Figure 9.

5. Experimental Results and Analysis

5.1. Experimental Settings. Parameter settings of ECOC algorithm are as follows: OVO, DECOE as encoding method, LR (logistic regression), SVM (support vector machine), Bayes as base classifier, and soft decoding as decoding method. The remaining parts adopt the default parameter settings.

Parameter settings of the GA are as follows: The number of individuals in the population is 80. The crossover probability is 0.8. The mutation probability is 0.1; and the maximum number of iterations is 50 generations. Parameter

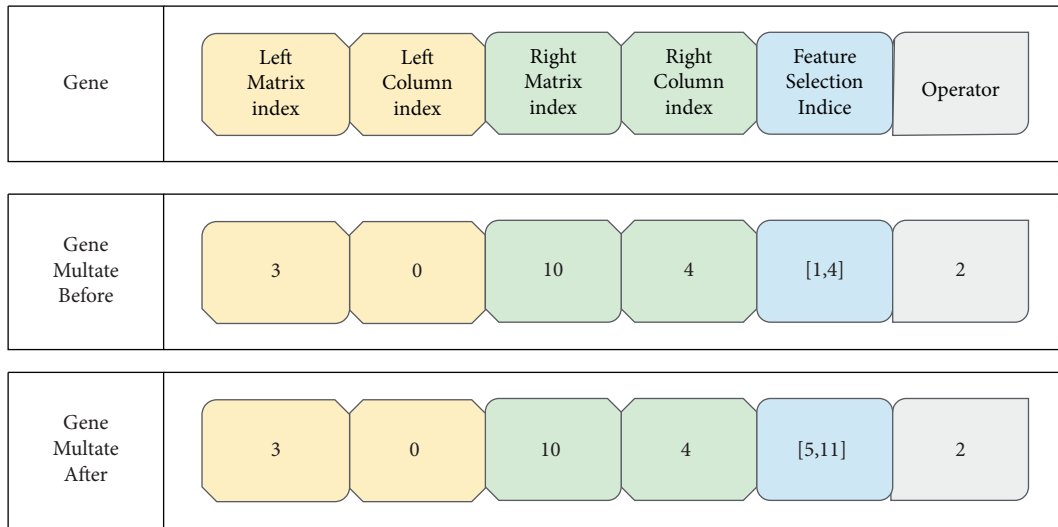


FIGURE 8: Choose feature selection sequence to mutate.

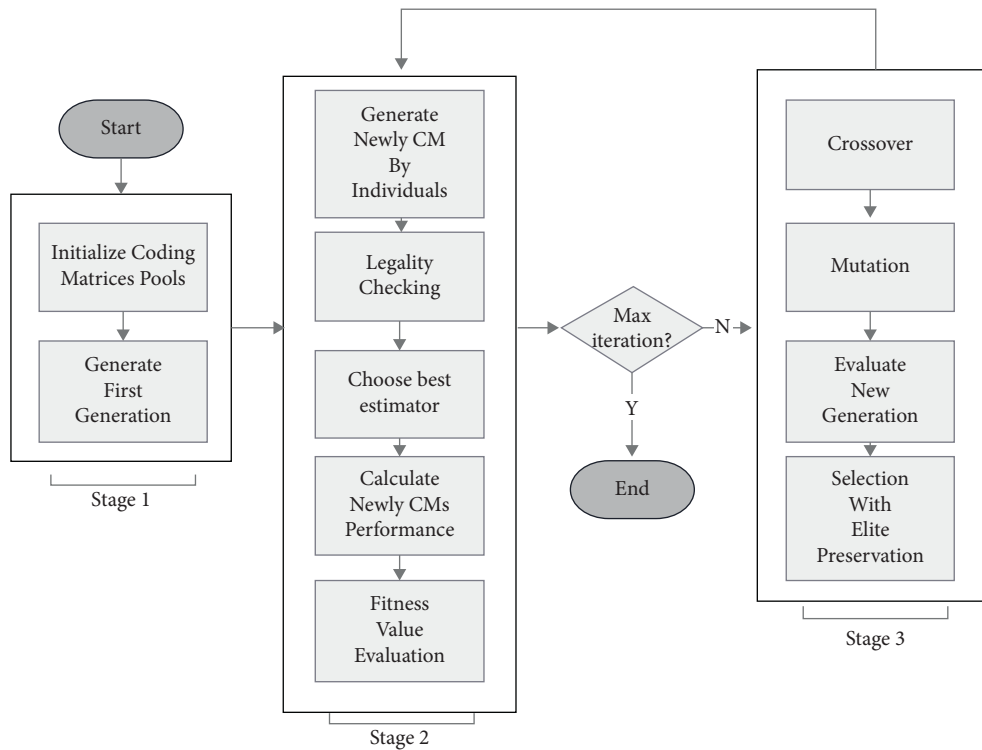


FIGURE 9: Flowchart of the proposed TBCGA-ECOC algorithm.

settings for both Random Forest and XGBoost use the default parameters in the scikit-learn library.

All the experiments employ the 5-fold crossover validation method. The average accuracy value is used as the evaluation index of the classification performance and the fitness in the GA.

5.2. Experimental Results

5.2.1. Comparison with the Classical Ensemble Learning Algorithms. TBCGA-ECOC proposed in this paper is

compared with Random Forest and XGBoost, two classical ensemble learning algorithms in machine learning. As shown in Figure 10, the TBCGA-ECOC significantly improves the accuracy of predicting students' performance compared with the two classic algorithms. Therefore, we conclude that the TBCGA-ECOC offers better performance compared to the traditional ensemble learning algorithms.

5.2.2. Comparison with the ECOC Algorithms. On the basis of the ECOC algorithm, the TBCGA-ECOC algorithm utilizes the GA-based feature selection method and base

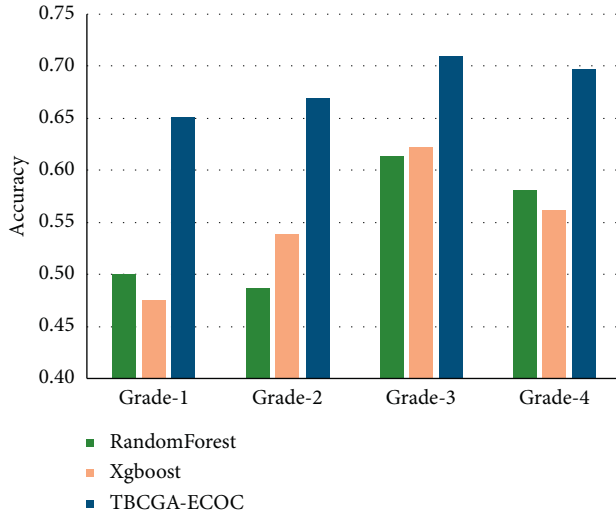


FIGURE 10: Comparison among TBCGA-ECOC and classic ensemble learning algorithms.

classifier selection to improve the prediction performance of the algorithm further. The comparative experimental results of the two classes of algorithms will be offered.

Table 5 reports the experimental results on different datasets when the OVO coding method is employed. As for grade-1 dataset: the accuracy of the TBCGA-ECOC algorithm increases about 3.53% and 3.58% over the results of the ECOC (LR) and ECOC (Bayes) algorithms, respectively; and it is pretty close to that obtained from the ECOC (SVM) algorithm. For the other three datasets, the TBCGA-ECOC algorithm results are better than those of ECOC (SVM), ECOC (LR), and ECOC (Bayes) algorithms. Among them, the algorithm results of the grade-2 dataset have increased by about 2.01%, 1.43%, and 7.11% based on the three algorithm results of ECOC (SVM), ECOC (LR), and ECOC (Bayes). The grade-3 dataset's accuracy is enhanced by 0.29%, 3.82%, and 13.15%. As for the grade-4 dataset, it also has an improvement of 1.42%, 2.53%, and 9.85%, respectively. The above results demonstrate that the proposed algorithm outperforms classical ECOC algorithms on all four datasets and shows significant improvement.

Similar results are also observed in experiments by using the DECOC encoding method. As shown in Table 6, the experimental results on different datasets are as follows.

Compared with the three algorithm results of ECOC (SVM), ECOC (LR), and ECOC (Bayes), the overall result of the TBCGA-ECOC algorithm has been improved. For example, the results of the grade-1 dataset are optimized by 1.97%, 4.42%, and 1.23%; and the grade-2 dataset's results are improved by about 3.17%, 3.17%, and 7.66%, respectively, based on other algorithms. For the two datasets, grade-3 and grade-4, the algorithm's results slightly rise on the basis of ECOC (SVM) and ECOC (LR) by 0.29% and 2.64% as well as 0.84% and 2.56%, respectively. Meanwhile, comparing the ECOC (Bayes) algorithm results, there is a significant enhancement, with an increase of 11.97% and 10.43%, respectively.

TABLE 5: Comparison of TBCGA-ECOC and ECOC (OVO)

	SVM	Logi	Bayes	TBCGA-ECOC
Grade 1	0.662 ± 0.330	0.622 ± 0.049	0.622 ± 0.053	0.657 ± 0.035
Grade 2	0.701 ± 0.022	0.707 ± 0.032	0.650 ± 0.071	0.721 ± 0.034
Grade 3	0.667 ± 0.022	0.631 ± 0.061	0.538 ± 0.032	0.667 ± 0.014
Grade 4	0.683 ± 0.032	0.672 ± 0.036	0.599 ± 0.033	0.697 ± 0.022

TABLE 6: Comparison of TBCGA-ECOC and ECOC (DECOC).

	SVM	Logi	Bayes	TBCGA-ECOC
Grade 1	0.638 ± 0.017	0.613 ± 0.031	0.645 ± 0.061	0.657 ± 0.035
Grade 2	0.690 ± 0.011	0.690 ± 0.028	0.645 ± 0.079	0.721 ± 0.034
Grade 3	0.667 ± 0.013	0.643 ± 0.034	0.550 ± 0.040	0.670 ± 0.014
Grade 4	0.689 ± 0.029	0.672 ± 0.050	0.593 ± 0.040	0.697 ± 0.022

Furthermore, it is observed that the accuracy of the algorithm on the grade-1 dataset is slightly low. The possible reasons are that the characteristic dimensions of the grade-1 dataset are fewer than those of the other three datasets, and the data have fewer dimensions. So the evolution is excessive under the same algorithm and parameters, and the overfitting phenomenon appears. Thus the accuracy is reduced to some extent. But, overall, TBCGA-ECOC improves to various degrees over ECOC accuracy on all four datasets.

These results conclude that the accuracy obtained from TBCGA-ECOC using different datasets or coding methods is higher than that from ECOC algorithms. Thus, the TBCGA algorithm based on GA feature selection and classifier traverse has excellent performance.

5.2.3. Comparison of Different Methods considering Prerequisite Courses. We tried to add the results of the prerequisite course to the dataset as a new feature; and, after data alignment, which can be achieved by eliminating the data missing features of prerequisite courses, we generated three new datasets with prerequisite courses and three without prerequisite courses corresponding to levels grade-2, grade-3, and grade-4.

To begin with, we utilize Random Forest and XGBoost, two traditional ensemble algorithms, to analyze and predict the six datasets with and without prerequisite courses. Figure 11 plots the prediction results.

There are six cases in total, and the prediction methods with prerequisite courses can obtain better results than those without prerequisite courses in 5 cases of 6 cases.

The two algorithms correspond to three grades, with six examples in total. In the five samples, the predicted results of the dataset with prerequisite courses were higher than or equal to those without prerequisite courses.

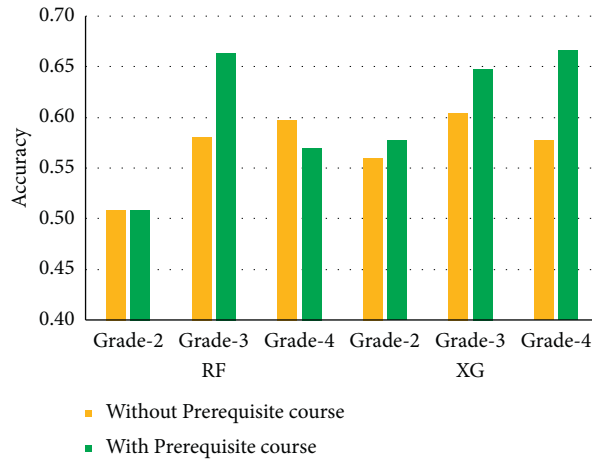


FIGURE 11: Feature contribution analysis on prerequisite courses for Random Forest and XGBoost.

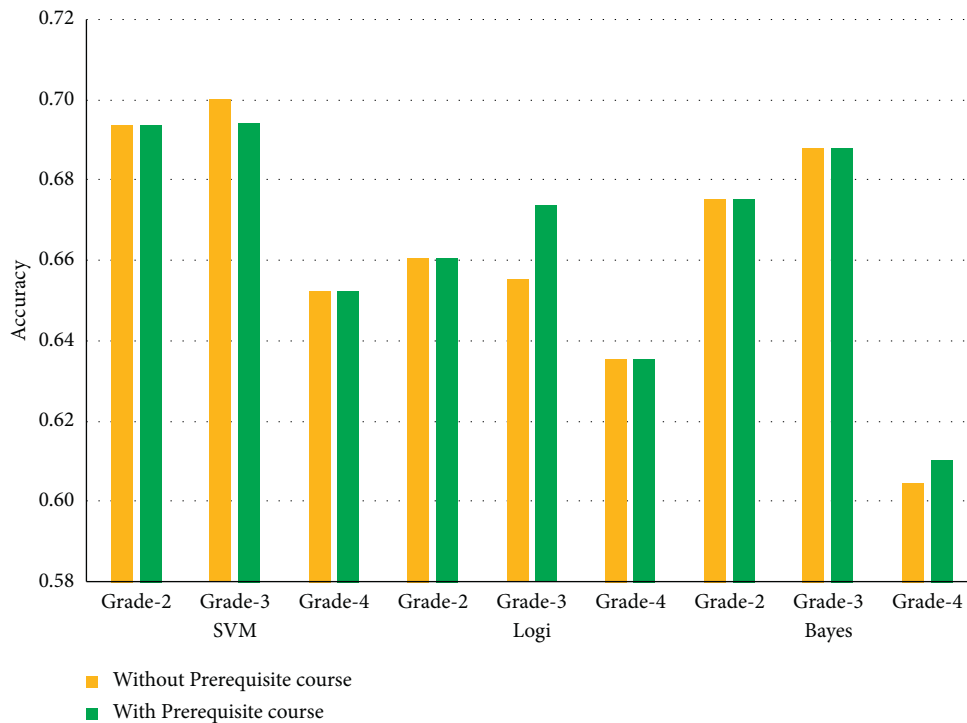


FIGURE 12: Feature contribution analysis on prerequisite courses for the ECOC (OVO) method.

In addition, the classical ECOC algorithm and TBCGA-ECOC algorithm are tested in the two kinds of datasets, and the results are listed in Figures 12–14.

When adopting the ECOC algorithm with the OVO coding method, three classifiers (SVM, Logi, and Bayes) correspond to three different datasets, for 9 cases in total. The results of the dataset incorporating the characteristics of prerequisite courses win or flat in 8 samples, as shown in Figure 12. A similar phenomenon is found when using the ECOC algorithm with the DECOC encoding method, with 8 of the 9 samples winning, as shown in Figure 13. Furthermore, when using the TBCGA-ECOC algorithm, the

dataset with prerequisite courses can get better results in all the cases, shown in Figure 14.

In general, the algorithm that adds the prerequisite courses as a new feature cannot consistently achieve better results compared to the original algorithm. Still, the performance is usually better than the average performance of the original algorithm, and it has good robustness. Therefore, it can be seen that the characteristics of the prerequisite course are helpful to the prediction of grades and have a better classification effect. Moreover, as shown in Figure 15, the prediction accuracy of the TBCGA-ECOC algorithm with the features of prerequisite

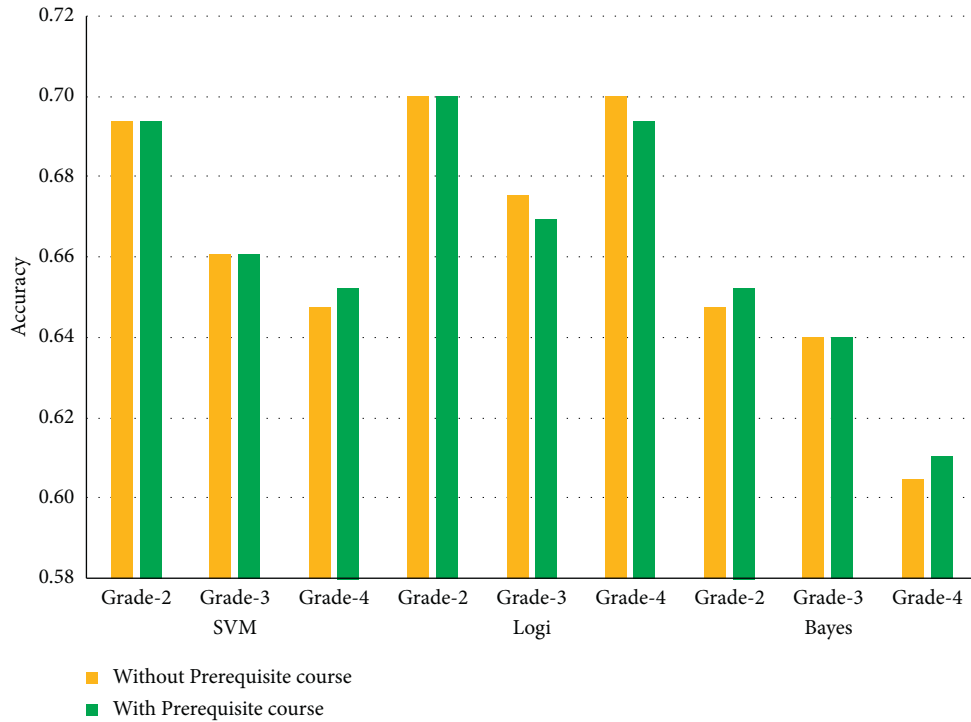


FIGURE 13: Feature contribution analysis on prerequisite courses for the ECOC (DECOC) method.

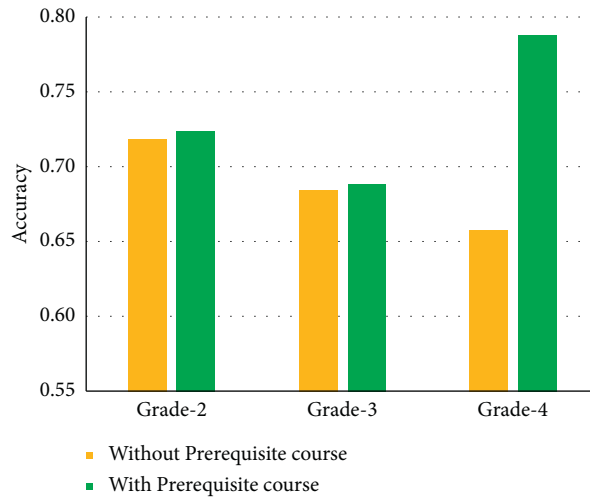


FIGURE 14: Feature contribution analysis on prerequisite courses for the TBCGA-ECOC.

courses added is also superior to those of the other algorithms on the whole.

5.2.4. Time Complexity. The running times of TBCGA-ECOC for grade-1, grade-2, grade-3, and grade-4 levels are 1826 s, 2122 s, 2025 s, and 2778 s, respectively. Based on the population iteration with high algorithm complexity, the GA algorithm consumes much time, while Random Forest and XGBoost, the classical ensemble learning algorithm models, are relatively simple and can complete training within 5 s. Although ECOC employs

different base classifiers and different coding methods, its algorithm’s complexity is also low.

The numbers of samples and features in the four datasets are different, so the computation times the algorithms need are different. The higher the feature dimension, the greater the solution space available for search. The grade-4 dataset with the highest dimension of features (131) and biggest number of samples (119) needs the longest running time, while the grade-1 dataset with the lowest dimension of features (110) and the smallest number of sample (82) needs the shortest running time. Finally, the grade-2 and grade-3 datasets need identical running times with similar

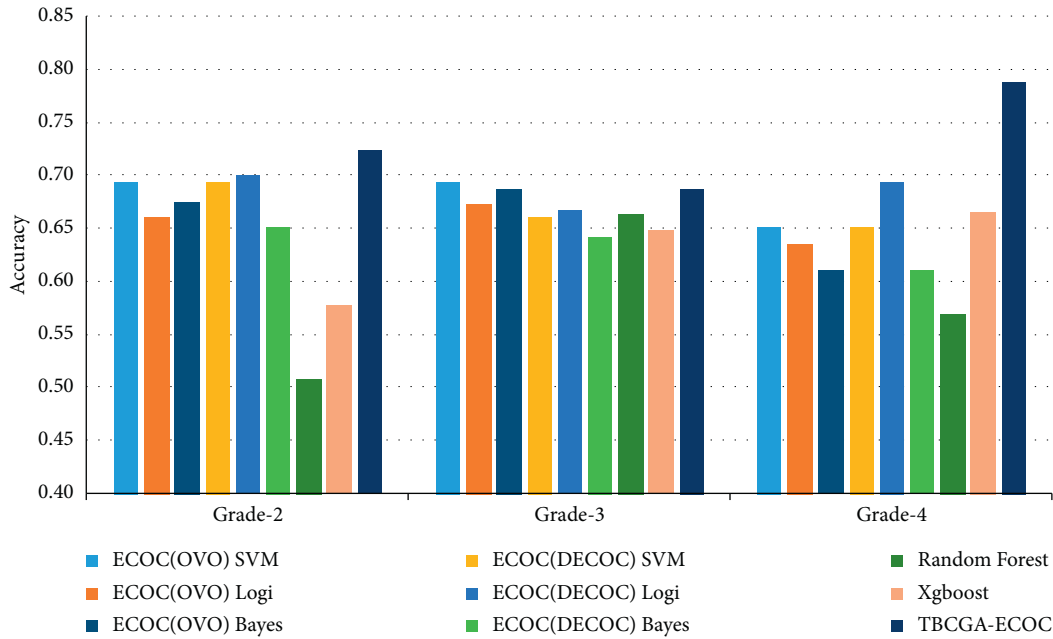


FIGURE 15: Comparison of TBCGA-ECOC and other methods with prerequisite courses feature.

dimensions and sample sizes. So the running time is less than that of the grade-4 dataset but more than that of the grade-1 dataset.

6. Conclusion

In this paper, we propose a novel ECOC multiclassification framework to predict students' grades. We try to add the prerequisite course as the new feature to improve the performance of the algorithm. The findings can be used to evaluate students' performance, providing valuable information for improving teaching in order to optimize individualized teaching, improve student performance, implement instructional interventions, and adjust the pace of teaching when necessary. The main conclusions are as follows:

- (1) Experimental results show that the proposed TBCGA-ECOC algorithm can obtain higher accuracy compared to several classical ECOC algorithms and the traditional ensemble learning algorithms, i.e., XGBoost and the Random Forest.
- (2) The prediction performance of various methods has been slightly or significantly improved by adding the grade of the prerequisites courses, i.e., *C language* and *Data Structure*, as the new features. On the contrary, the proposed method can provide valuable advice for curriculum and teaching planning. According to the predicted results, we can recommend suitable prerequisites to students.

In future work, we can follow the line of this work. As for the dataset, we can collect other multisource learning track data or add other prerequisite courses. As for the method, the ECOC algorithm can be improved, especially its adaptability with high-dimensional small sample data; or the

selection scheme of hybrid binary classifiers can be optimized so that the better binary classifiers can be selected to enhance accuracy.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Authors' Contributions

Ling-qing Chen and Mei-ting Wu contributed equally to this work and should be regarded as joint first authors.

Acknowledgments

This work was supported in part by the Scientific Research Program of Fujian Bureau of Education, China (no. JAT200266), the Opening Fund of Digital Fujian Big Data Modeling and Intelligent Computing Institute, and the Scientific Research Foundation of Jimei University, China (no. F020803).

References

- [1] A. M. F. Yousef and T. Sumner, "Reflections on the last decade of MOOC research," *Computer Applications in Engineering Education*, vol. 29, no. 4, pp. 1–18, 2020.
- [2] K. L.-M. Ang, F. L. Ge, and K. P. Seng, "Big educational data & analytics: survey, architecture and challenges," *IEEE Access*, vol. 8, pp. 116392–116414, 2020.

- [3] Z. Xie, "Modelling the dropout patterns of MOOC learners," *Tsinghua Science and Technology*, vol. 25, no. 3, pp. 313–324, 2020.
- [4] Z. Xu, H. Yuan, and Q. Liu, "Student performance prediction based on blended learning," *IEEE Transactions on Education*, vol. 64, no. 1, 2020.
- [5] K. F. Hew, X. Hu, C. Qiao, and Y. Tang, "What predicts student satisfaction with MOOCs: a gradient boosting trees supervised machine learning and sentiment analysis approach," *Computers & Education*, vol. 145, Article ID 103724, 2020.
- [6] H. Hu, G. Zhang, W. Gao, and M. Wang, "Big data analytics for MOOC video watching behavior based on spark," *Neural Computing & Applications*, vol. 32, no. 11, pp. 6481–6489, 2019.
- [7] C.-H. Yu, J. Wu, M.-C. Liu, and A.-C. Liu, "Adopting software product lines to implement an efficient learning analytics framework in MOOCs," *Journal of Information Science and Engineering*, vol. 37, no. 1, pp. 139–155, 2021.
- [8] L. Zhang, Y. Xuan, and H. Zhang, "Construction and application of SPOC-based flipped classroom teaching mode in installation engineering costcurriculum based on OBE concept," *Computer Applications in Engineering Education*, vol. 28, no. 6, pp. 1503–1519, 2020.
- [9] Y. Wang, C. Dong, and X. Zhang, "Improving MOOC learning performance in China: an analysis of factors from the TAM and TPB," *Computer Applications in Engineering Education*, vol. 28, no. 6, pp. 1421–1433, 2020.
- [10] J. Gardner and C. Brooks, "Student success prediction in MOOCs," *User Modeling and User-Adapted Interaction*, vol. 28, no. 2, pp. 127–203, 2018.
- [11] A. Onan and M. A. Toçoğlu, "Weighted word embeddings and clustering-based identification of question topics in MOOC discussion forum posts," *Computer Applications in Engineering Education*, vol. 29, no. 4, 2020.
- [12] M. A. Toçoğlu and A. Onan, "Sentiment analysis on students' evaluation of higher' educational institutions," *Advances in Intelligent Systems and Computing*, Springer, Germany, pp. 1693–1700, 2021.
- [13] O. Aytuğ, "Sentiment analysis on Twitter based on ensemble of psychological and linguistic feature sets," *Balkan Journal of Electrical and Computer Engineering*, vol. 6, no. 2, pp. 69–77, 2018.
- [14] A. Onan, "Mining opinions from instructor evaluation reviews: a deep learning approach," *Computer Applications in Engineering Education*, vol. 28, no. 1, pp. 117–138, 2020.
- [15] A. Onan, "Sentiment analysis on product reviews based on weighted word embeddings and deep neural networks," *Concurrency and Computation: Practice and Experience*, Article ID e5909, 2020.
- [16] J. Xu, K. H. Moon, and M. Van Der Schaar, "A machine learning approach for tracking and predicting student performance in degree programs," *IEEE Journal of Selected Topics in Signal Processing*, vol. 11, no. 5, pp. 742–753, 2017.
- [17] R. L. Ulloa-Cazarez, C. López-Martín, A. Abran, and C. Yáñez-Márquez, "Prediction of online students performance by means of genetic programming," *Applied Artificial Intelligence*, vol. 32, no. 9–10, pp. 858–881, 2018.
- [18] B. Xu and D. Yang, "Motivation classification and grade prediction for MOOCs learners," *Computational Intelligence and Neuroscience*, vol. 2016, Article ID 2174613, 7 pages, 2016.
- [19] Y. Meier, J. Xu, O. Atan, and M. Van der Schaar, "Predicting grades," *IEEE Transactions on Signal Processing*, vol. 64, no. 4, pp. 959–972, 2015.
- [20] J. Qiu, J. Tang, T. X. Liu et al., "Modeling and predicting learning behavior in MOOCs," in *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*, pp. 93–102, New York, NY, USA, February 2016.
- [21] M. Zhang, J. Zhu, Z. Wang, and Y. Chen, "Providing personalized learning guidance in MOOCs by multi-source data analysis," *World Wide Web*, vol. 22, no. 3, pp. 1189–1219, 2019.
- [22] C.-H. Yu, J. Wu, and A.-C. Liu, "Predicting learning outcomes with MOOC clickstreams," *Education Sciences*, vol. 9, no. 2, p. 104, 2019.
- [23] A. A. Mubarak, H. Cao, W. Zhang, and W. Zhang, "Visual analytics of video-clickstream data and prediction of learners' performance using deep learning models in MOOCs' courses," *Computer Applications in Engineering Education*, vol. 29, no. 4, pp. 710–732, 2020.
- [24] Y. Wen, Y. Tian, B. Wen, Q. Zhou, G. Cai, and S. Liu, "Consideration of the local correlation of learning behaviors to predict dropouts from MOOCs," *Tsinghua Science and Technology*, vol. 25, no. 3, pp. 336–347, 2020.
- [25] M. Hussain, W. Zhu, W. Zhang, S. M. R. Abidi, and S. Ali, "Using machine learning to predict student difficulties from learning session data," *Artificial Intelligence Review*, vol. 52, no. 1, pp. 381–407, 2018.
- [26] Q. Chen, X. Yue, X. Plantaz et al., "ViSeq: visual analytics of learning sequence in massive open online courses," *IEEE Transactions on Visualization and Computer Graphics*, vol. 26, no. 3, pp. 1622–1636, 2018.
- [27] R. Cobos and J. C. Ruiz-Garcia, "Improving learner engagement in MOOCs using a learning intervention system: a research study in engineering education," *Computer Applications in Engineering Education*, vol. 29, no. 4, 2020.
- [28] S. García-Molina, C. Alario-Hoyos, P. M. Moreno-Marcos, P. J. Muñoz-Merino, I. Estévez-Ayres, and C. Delgado Kloos, "An algorithm and a tool for the automatic grading of MOOC learners from their contributions in the discussion forum," *Applied Sciences*, vol. 11, no. 1, p. 95, 2021.
- [29] X. Chen, J. Zheng, Y. Du, and M. Tang, "Intelligent course plan recommendation for higher education: a framework of decision tree," *Discrete Dynamics in Nature and Society*, vol. 2020, Article ID 7140797, 11 pages, 2020.
- [30] M. Sun, K. Liu, Q. Wu, Q. Hong, B. Wang, and H. Zhang, "A novel ECOC algorithm for multiclass microarray data classification based on data complexity analysis," *Pattern Recognition*, vol. 90, pp. 346–362, 2019.
- [31] T. G. Dietterich and G. Bakiri, "Solving multi-class learning problems via error-correcting output codes," *Journal of Artificial Intelligence Research*, vol. 2, pp. 263–286, 1994.
- [32] M. Galar, A. Fernández, E. Barrenechea, H. Bustince, and F. Herrera, "NMC: nearest matrix classification - a new combination model for pruning one-vs-one ensembles by transforming the aggregation problem," *Information Fusion*, vol. 36, pp. 26–51, 2017.
- [33] K. Crammer and Y. Singer, "On the learnability and design of output codes for multi-class problems," *Machine Learning*, vol. 47, no. 2, pp. 201–233, 2002.
- [34] L. Zhou, Q. Wang, and H. Fujita, "One versus one multi-class classification fusion using optimizing decision directed acyclic graph for predicting listing status of companies," *Information Fusion*, vol. 36, pp. 80–89, 2017.
- [35] F. Masulli and G. Valentini, "Effectiveness of error correcting output coding methods in ensemble and monolithic learning machines," *Formal Pattern Analysis & Applications*, vol. 6, no. 4, pp. 285–300, 2004.

- [36] X.-N. Ye, K.-H. Liu, and S.-T. Liong, "A ternary bitwise calculator based genetic algorithm for improving error correcting output codes," *Information Sciences*, vol. 537, pp. 485–510, 2020.
- [37] Y.-P. Zhang, X.-N. Ye, K.-H. Liu, and J.-F. Yao, "A novel multi-objective genetic algorithm based error correcting output codes," *Swarm and Evolutionary Computation*, vol. 57, Article ID 100709, 2020.
- [38] M. A. Bagheri, Q. Gao, and S. Escalera, "A genetic-based subspace analysis method for improving error-correcting output coding," *Pattern Recognition*, vol. 46, no. 10, pp. 2830–2839, 2013.
- [39] L. I. Kuncheva and L. C. Jain, "Designing classifier fusion systems by genetic algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 4, no. 4, pp. 327–336, 2000.
- [40] D. Dutta, J. Sil, and P. Dutta, "Automatic clustering by multi-objective genetic algorithm with numeric and categorical features," *Expert Systems with Applications*, vol. 137, pp. 357–379, 2019.

Research Article

Predicting Learning Behavior Using Log Data in Blended Teaching

Shu-Tong Xie ^{1,2}, Zong-Bao He,¹ Qiong Chen ³, Rong-Xin Chen ¹, Qing-Zhao Kong,⁴
and Cun-Ying Song⁵

¹School of Computer Engineering, Jimei University, Xiamen 361021, China

²Digital Fujian Big Data Modeling and Intelligent Computing Institute, Jimei University, Xiamen 361021, China

³College of Computer and Information Sciences, Fujian Agriculture and Forestry University, Fuzhou 350002, China

⁴School of Science, Jimei University, Xiamen 361021, China

⁵School of Foreign Languages, Jimei University, Xiamen 361021, China

Correspondence should be addressed to Qiong Chen; qchen@fafu.edu.cn

Received 10 May 2021; Revised 15 June 2021; Accepted 14 August 2021; Published 24 August 2021

Academic Editor: Jiwei Huang

Copyright © 2021 Shu-Tong Xie et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Online and offline blended teaching mode, the future trend of higher education, has recently been widely used in colleges around the globe. In the article, we conducted a study on students' learning behavior analysis and student performance prediction based on the data about students' behavior logs in three consecutive years of blended teaching in a college's "Java Language Programming" course. Firstly, the data from diverse platforms such as MOOC, Rain Classroom, PTA, and cnBlog are integrated and preprocessed. Secondly, a novel multiclass classification framework, combining the genetic algorithm (GA) and the error correcting output codes (ECOC) method, is developed to predict the grade levels of students. In the framework, GA is designed to realize both the feature selection and binary classifier selection to fit the ECOC models. Finally, key factors affecting grades are identified in line with the optimal subset of features selected by GA, which can be analyzed for teaching significance. The results show that the multiclass classification algorithm designed in this article can effectively predict grades compared with other algorithms. In addition, the selected subset of features corresponding to learning behaviors is pedagogically instructive.

1. Introduction

With the rapid advancement of Internet technology, blended teaching mode has emerged in recent years. Different from the previous online teaching mode, blended teaching includes an integration of learners' online learning with traditional face-to-face classroom learning, which not only can guarantee the degree of interaction between teachers and students but also can ensure the efficiency of students' online learning owing to the supervision of teachers. Through analyzing students' online learning behavior and classroom performance, teachers can gain insight into the comprehensive information of students, making it possible to tailor their teaching to students' needs to further improve the teaching quality. Therefore, there is a need to study how to fully exploit the data of students' online and offline learning behavior to effectively guide students' individualized learning.

At present, most of the research work is typically based on students' learning behavior data to predict whether students can pass examinations, obtain certificates, or dropout of classes, etc., which are essentially binary class problems. Besides, much of the work is done to extract coarse-grained features from data for prediction. In this work, however, we particularly focus on the prediction of students' performance in four categories (excellent, good, passed, and failed) upon their learning behavior data. Specifically, we add a genetic algorithm- (GA-) based feature selection and binary classifier selection to the error correct output codes (ECOC) framework to filter the learning behavior features of the original data, in order to improve the accuracy of multiclassification in many ways. According to the fine-grained results of feature selection and correlation analysis, the learning features that have a more significant impact on students' final grades are selected from their

learning behaviors for analysis, which is conducive to teachers' individualized teaching.

The structure of this article is organized as follows. Section 2 discusses the related work. Section 3 primarily gives a description of the course and data involved. Section 4 briefly introduces ECOC algorithm and GA as well as details the design and implementation of the ECOC-GA algorithm. Section 5 presents the experimental setup and the comparison combined with analysis of the experimental results and then puts forward some implications for teaching. We conclude the whole work in final section.

2. Literature Review

In this section, we will briefly review the research and application of machine learning and mining technology in online learning. Ang et al. conducted a comprehensive survey of the research related to big educational data that has emerged in the past few years [1]. They reviewed the overview and classification of big educational data research, data collection and mining efforts for big educational data, methodological and technical issues involved in big educational data, and challenges faced by big educational data. Related research studies showed that prediction of learning effectiveness had an important research value on the education field, thereby receiving much attention. Basing on a dataset collected from XuetangX, one of the largest MOOCs from China, Qiu et al. proposed a latent dynamic factor graph (LadFG) to predict students' homework completion and whether they can successfully pass the examinations and obtain the certificates [2]. By analyzing students' activity logs on the MOOCs, Xu and Yang summarized various learning motivations of students and designed a classification algorithm based on support vector machine (SVM) to predict whether students can obtain a certificate [3]. Working with multisource heterogeneous data from two courses by the Peking University on Coursera, *Data Structures and Algorithms and Introduction to Computing*, Zhang et al. analyzed students' learning content, identified important concepts in the course, assessed students' knowledge states through the quiz, and designed algorithms to predict student dropout in MOOCs [4]. Yu et al. identified seven types of cognitive participation models of learners based on their video clickstream records and built practical machine learning models using K-nearest neighbors (KNN), SVM, and artificial neural networks (ANN) to make predictions about whether students can pass the course examinations [5]. In order to significantly improve the teaching efficiency of traditional courses and online MOOCs, Meier et al. used history of teaching data of the course (mainly including assignments, quizzes, midterm examination, etc.) to predict the likely performance (good/poor) of students in subsequent courses so that teachers can initiate interventions promptly for students performing poorly [6]. Ulloa-Cazarez et al. proposed a genetic programming (GP) algorithm to predict whether students can pass the final examination [7]. Trying to identify students who have difficulty in course learning, Xu et al. developed a machine learning algorithm with a bilayered structure consisting of multiple base

predictors and a cascade of ensemble predictors [8]. This algorithm could be used to make predictions based on students' progressive performance states. Additionally, a clustering method based on latent factor models and probabilistic matrix factorization was proposed to discover the correlation among the courses to further construct more effective base predictors. The results suggested that the method had a positive impact on assessing student performance and was able to provide implications of teaching intervention for instructors. Hussain et al. used machine learning algorithms including ANN, SVM, Logistic Regression (LR), Naive Bayes (NB), and Decision Trees (DT) to analyze data recorded by the TEL system. The model was trained according to the data of student performance from the previous week and then tested on the data from the following week [9]. Hence, the differences in performance among various algorithms could be compared. The results showed that ANN and SVM achieved higher accuracy than other algorithms. Overall, the method employed was highly real time and especially helpful in allowing teachers to identify underprepared students and facilitate interventions before the next week's class.

In addition to the research on the prediction of learning effectiveness, there is also an increasing body of work in the field of students' behavioral patterns and the influencing factors of learning. Basing on the Spark platform, Hu et al. proposed a method for analyzing students' video watching behavior in MOOCs and validated it with the data of the caux platform [10]. The experimental results indicated that the method could quickly and accurately analyze the characteristics of video watching behavior, which is conducive to explaining the time-distributed video watching behavior in MOOCs. Besides, it could also guide instructors to determine a reasonable video length to draw more students' attention and reduce their dropout rate. Theoretically guided by Moore's theory of transactional distance, Hew et al. adopted approaches incorporating supervised learning, sentiment analysis, and hierarchical linear modeling to analyze the cognitive features of 6,393 students in 249 random MOOCs, with MOOC learners' satisfaction as a measure of success [11]. The results revealed that teacher, content, assessment, and schedule had significant effects on student satisfaction. Onan et al. developed a document-clustering model based on weighted word embedding to identify question topics on posts of MOOC forum [12]. Later, Onan designed a long short-term memory networks (LSTM) to classify the sentiment for about 70,000 MOOC reviews, which can achieve a high classification accuracy [13]. In addition, sentiment analysis on both Twitter data and students' evaluation data are also investigated recently [14–16]. Due to teaching inertia and difficulty of fixed curriculum setting in coping with the IT field where knowledge is updated rapidly, Chen et al. proposed a data-based framework to evaluate the relationship between taking courses and getting employment through the DT expression [17]. Based on it, a computer course group recommendation algorithm was proposed to provide a basis for optimal course configuration. The results showed that the course optimization and recommendation results based on this method had a significant contribution to student

employment. Yousef and Sumner conducted a comprehensive analysis of more than 200 articles on MOOC research and discussed the main research findings and research directions on MOOC in recent years around the issues of classification of MOOC, MOOC learners, communication between MOOC providers and participants, and assessment for students' learning behavior [18]. Mubarak et al. used a twofold analysis method combining visual and predictive analyses to make an initial prediction of learners' performance through the analysis results of data visualization and then built the RNN-LSTM model for learners' behavior in video clickstreams to improve the predictability of learner performance [19]. Zhang et al. constructed a SPOC-based flipped classroom teaching model by integrating the OBE concept, and the results confirmed that the model in small classes had equally ideal effects on both teachers and students [20]. Chen et al. introduced and evaluated an interactive visual analytics system called ViSeq, showing that it could visualize the learning sequences of different learner groups to aid users in exploring them of MOOCs at multiple levels of granularity and comprehending the underlying patterns behind the learning sequences [21]. Combining TAM with TPB and collecting data from questionnaires completed by university learners, Wang et al. constructed a theoretical model of MOOC learning performance mechanisms [22]. The analysis of learning groups' characteristics revealed that factors such as perceived usefulness, learning attitudes, subjective norms, and perceived behavioral control had disproportionate impacts on learning performance, while perceived ease of use was not a crucial factor. The results provided a scientific basis for MOOC instructional design and instructional resources allocation. Cobos and Ruiz-Garcia proposed a learning intervention system called edX-LIS for providing MOOC learners with information about their progress regularly and making suggestions by their performance [23]. Contrast experiments had shown that the intervention strategies provided by this system had a positive impact on learners' motivation, persistence, and engagement. García-Molina et al. proposed an algorithm for automatic scoring of learners' performance in MOOC forums [24]. Simultaneously, they collected data for an exploratory study that proposed three schemes with different input parameters. The results showed a moderate positive correlation between the forum grades provided by the algorithm and the grades obtained through the summative assessment activity of MOOC. Xie conducted a survival analysis for the video viewing duration of MOOCs and developed a mathematical model to understand the evolutionary mechanisms underlying the characteristics of students' video viewing behavior [25]. The study also explored the potential role of memory in the complexity of learning behaviors. Wen et al. conducted a detailed analysis of learning behavior patterns of MOOC learners and proposed a novel simple feature matrix for maintaining the local correlation information of learning behavior [26]. The study found that learners often exhibited similar learning behaviors over consecutive days. By making use of it, a brand new convolutional neural network (CNN) model was also proposed to improve the accuracy of dropout prediction. Yu et al. applied the software product line (SPL) in software engineering

techniques to a framework for data analysis, which made the process of data analysis reusable [27]. A practical machine learning model was constructed on the basis of this framework to predict learning outcomes through students' learning behavior. The framework retained data collection, data cleaning, feature extraction, and model prediction as core components. As a result, a complete and reusable learning behavior analysis system came into being.

3. Data Description

3.1. Curriculum Setting. Our algorithm was applied to data collected from a blended course of *Java Language Programming* in a college for three consecutive years. The course aimed at understanding the basic knowledge of Java and learning Java programming techniques, with a total of 68 class hours. While participating in traditional courses in classroom, students also studied online through e-learning platforms, such as Rain Classroom, MOOC, PTA, and Blog. They would have a final examination at the end of the semester.

3.2. Data Description. In this article, we collected learning data generated by students in three grades of a college while participating in the course (the three different grades are represented by grade 1, grade 2, and grade 3). Among them, 94, 127, and 130 students in grades 1, 2, and 3 participated in the course, respectively. The data were mainly obtained from four platforms: MOOC platform, Rain Classroom, Programming Teaching Assistant (PTA), and cnBlog. The details are summarized in Table 1.

3.2.1. Data of MOOC. The data of MOOC platform mainly includes 102 lecture videos, 6 peer grading assignments, 8 chapter quizzes, forum discussions, online examination, and other types of learning activity data. The specific fields are shown in Table 2.

3.2.2. Data of Rain Classroom. The Rain Classroom data mainly contains two categories: preclass preparation and classroom participation. 25 preclass preparation cases and 9 preclass preparation cases are included in the Rain Classroom data of grade 1; 27 preclass preparation cases and 16 classroom participation cases are included in the Rain Classroom data of grade 2; 26 preclass preparation cases and 15 classroom participation cases are included in the Rain Classroom data of grade 3. The detailed descriptions are listed in Table 3.

3.2.3. Data Integration. In addition to the data from MOOC platform and Rain Classroom platform, cnBlog assignments and PTA examinations are also set up to enrich the course content. Therefore, it is necessary to integrate MOOC data, Rain Classroom data, cnBlog data, and PTA data. The integrated data are merged and aligned with the data of students' final examination grades, which are classified into four levels: excellent, good, passed, and failed.

TABLE 1: Course data source.

Semester	Number of students	Data source
Fall semester 2017-2018 (grade 1)	94	Rain Classroom and cnBlog
Fall semester 2018-2019 (grade 2)	127	Rain Classroom and PTA
Fall semester 2019-2020 (grade 3)	130	MOOC, Rain Classroom, and PTA

TABLE 2: Data of MOOC.

Features	Explanation	Note
Student information	It is made up of student number, name, class, etc.	
Task response	Number of tasks completed/task completion percentage	
Number of discussions	Forum discussion	
Chapter learning times		Students can watch lectures multiple times
Video viewing duration		Duration (10.3 minutes)
SPOC-debugging.mp4		Duration (1.7 minutes)
...	There are 102 videos here	.
Introduction.mp4		Duration (1.7 minutes)
Number of posts/reply discussions		
Suggested score	Student scores suggested by the system	
Course video progress	Number of completed course videos/total number	
Chapter test progress		
Total performance	Students' online scores	Hundred mark
Five level (ABCDE)	Turn grades into levels	
Certificate issuing status		Yes/no
Peer assessment—1		Score
Submission of peer graded assignment	There are 6 peer grading	Yes/no
Participation in peer graded assignment	Assignments here	Yes/no
...		
Assignment grade of unit 1		Hundred-mark
Assignment submission time of unit 1	Time: YYYY-MM-DD hh:mm:ss	
...		
Assignment grade of unit 8	Details of the assignments and	
Assignment submission time of unit 8	Quizzes for a total of eight units	

TABLE 3: Data of Rain Classroom.

Features	Explanation	Note
Students' ID/name/number	Students' information	
Total score		
Attendance rate	Attendance times/total times	
Number of notices read		
Total pages viewed PPT		Preclass preparation
Total duration	Time for watching courseware	Preclass preparation
Completion time	Time to complete courseware	Preclass preparation
Score	Answer situation in courseware	Preclass preparation
Sign-in method		Classroom participation
Number of pop-ups	Total number of pop-ups sent	Classroom participation
Cumulative score		Classroom participation

Finally, after data filtering, feature transformation, and other preprocessing operations, the grade 1 dataset contains 82 samples in 138 dimensions; the grade 2 dataset includes 118 samples in 151 dimensions; the grade 3 dataset comprises 115 samples in 291 dimensions.

4. The Design of ECOC-GA

Classification is a common problem in machine learning, which can be further divided into binary classification and

multiclassification. In this work, the ECOC algorithm is utilized to integrate multiple binary classifiers to solve the multiclassification problem.

4.1. Introduction to ECOC. ECOC first came from the field of communication as a technique for correcting information transmission errors in networks. In the field of machine learning, ECOC algorithm essentially serves as an ensemble learning framework, using different base classifiers to learn

prior knowledge at different levels on dataset. Accordingly, the overall performance of models can be effectively improved for the mutual complementation of the classifiers [28–30].

ECOC algorithm mainly consists of three basic steps: the encoding step, the training step, and the decoding step [31]. The encoding strategies are mainly divided into data-independent encoding and data-dependent encoding. To be specific, data-independent encoding can be further divided into One-vs-All (OVA), One-vs-One (OVO) [32], Dense Random (DR) [33], and Sparse Random (SR) [34], whereas data-dependent encoding includes D-ECOC, ECOC-ONE, Forest-ECOC, and other types [35]. An ECOC algorithm guides the training and prediction of multiple binary classification models by generating a code matrix. Take the encoding process of the OVA matrix as an example, the matrix row vector represents a class and the column vector represents a binary classifier. As depicted in Figure 1, the first column of the code matrix $\{1, -1, -1, -1\}$ indicates the first column corresponding to the binary classifier h_1 treats the data of class $\{C_1\}$ as a positive group and the data of other classes as a negative group, and then the model h_1 can be trained utilizing the training data divided into two classes.

In the decoding stage, when predicting the output label of a certain sample, the prediction results returned by each dichotomizer will form a result vector. By calculating the distance between this result vector and the code word in the code matrix, the code word with the smallest distance between them is selected as the final prediction result.

4.2. Feature Selection Algorithm. The quality of data is the key to machine learning. If the original dataset has too many redundant features, it will not only cause serious interference to the whole learning process but also cause a waste of computation time and memory. Feature selection can select some of the most important features in the dataset, which enhances models' generalization ability and reduces overfitting, as well as improves the interpretability between features and prediction targets.

GA is a global algorithm to search the solution space and find the optimal solution. It is formed by simulating Darwin's natural selection theory and combining it with the theory of biological evolution in genetics [36]. Selection, crossover, and mutation constitute the genetic operator of GA. In addition, setting of the initial population, development of the individual coding strategy, design of the fitness function, design of the genetic operator, and setting of the control parameters form the core of GA. Related studies have proved that GA as a feature selection algorithm [37, 38] can select more relevant features in comparison with traditional algorithms, thus further improving the predictive performance of models.

4.3. Multiclassification Algorithm Based on ECOC and GA. Classical ECOC algorithms take all features as input variables and adopt the same kind of binary classifiers set to fit models and make predictions. In contrast, we designed a multiclassification algorithm based on ECOC and GA for

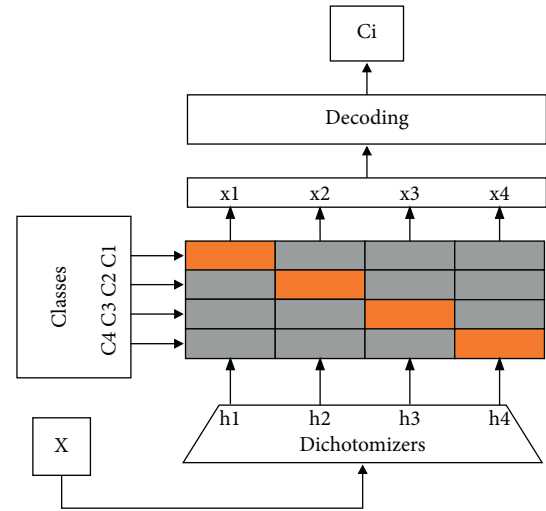


FIGURE 1: OVA code matrix for a four-class problem.

performance prediction, which uses a heterogeneous set of binary classifiers. GA is used not only for feature selection but also for binary classifier selection for each column in the code matrix. The specific steps of the algorithm are as follows:

- (1) Representation of chromosome: feature subset coding is the mapping of sample feature selection in the dataset. Hence, each feature corresponds to a binary number (1 represents selection of the feature, 0 represents non-selection of the feature). As a result, we can get n -bit binary chromosome codes (n is the dimension of training dataset). The base classifier sequence coding is the mapping of the selection of binary classifiers in each column of the code matrix. There are three optional base classifiers, and accordingly, each column corresponds to one trinary number and m -bit trinary chromosome codes appear (m is the number of columns in code matrix).
- (2) Setting of relevant parameters: initialization settings for various parameters such as the number of iterations of GA, population size, elite preservation rate, random selection rate, and mutation rate.
- (3) Population initialization: the chromosomes of each individual are coded by a random function. Each individual has two chromosomes. According to the chromosome coding sequence, some students' learning behaviors and the optimal set of base classifiers are all selected.
- (4) Selection: the selection operation of GA is divided into two main parts. Firstly, the top 20% of the most adapted individuals in current population are selected based on the elite preservation rate. Then, some of the less adaptive but surviving individuals are selected among the remaining individuals based on the random selection rate. The method is consistent with the randomness of natural selection, and it also allows the whole population to obtain a higher fitness value.

- (5) Crossover: we adopt the method of single-point crossover, randomly selecting two individuals from the population as parents and randomly selecting a position of the individual's chromosome as a crossover site. Using the crossover site as the intercept point, we intercept two parts of the parents, respectively, to reassemble two new individuals. Assuming a parent individual's chromosome is 1000,0010,0010,1010, the other is 0101,1010,0001,1110, and the crossover site is 2. Then, the two offspring individuals' chromosomes after crossover are 1100,0010,0010,1010 and 0001,1010,0001,1110, respectively. The population is replenished to keep the population size constant during the iteration.
- (6) Mutation: according to the mutation rate, one of the chromosomes of an individual is randomly selected for gene mutation to obtain a new individual and update the population.
- (7) Termination condition: the algorithm can be stopped when the number of iterations meets the requirement, and the result can be used as the current optimal solution.

We use an ECOC-based framework for models' training and prediction and adopt the 5-fold cross-validation scheme to evaluate the performances. The average value of the accuracy is calculated as the fitness to measure how much GA improves the performance of the overall multiclassification problem. The feature subset and base classifiers sequence with high fitness are selected to enter the next iteration, and the final individual with the highest fitness, i.e., containing the optimal feature subset and the optimal set of base classifiers, is obtained. The flowchart of the proposed ECOC-based algorithm (referred to as ECOC-GA) is shown in Figure 2.

5. Experimental Results and Analysis

5.1. The Setting of Experiments. The parameter of ECOC algorithm: encoding methods are OVO, DR, and DECOG; base classifiers are LR, SVM, and Bayes; decoding method is soft decoding; the rest are set with default parameters.

The parameter of GA: the number of individuals in the population is 85, the random selection rate is 0.5; the mutation rate of the feature subset is 0.03; the mutation rate of the base classifiers sequence is 0.08; the elite preservation rate is 0.2, and the maximum number of iterations is 65. In addition, Random Forest and XGBoost are provided by the scikit-learn library with default parameter settings.

In this work, all experiments are conducted using the 5-fold cross-validation scheme, and the mean value of accuracy is used as an evaluation index for the classification performance and fitness values in GA.

5.2. Comparison of Experimental Results

5.2.1. Comparison with Traditional Ensemble Learning Algorithms. There are many classical ensemble learning

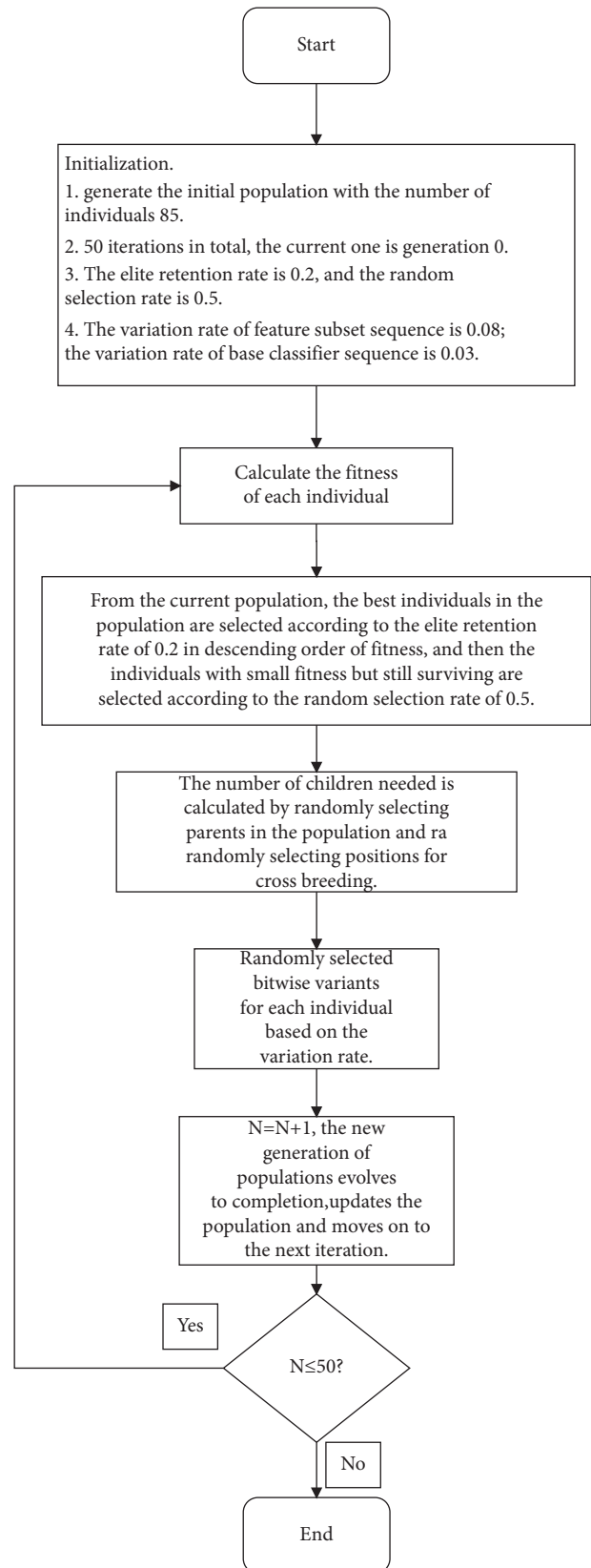


FIGURE 2: Flowchart of ECOC-GA algorithm.

algorithms in machine learning such as Random Forest and XGBoost. The comparison of the ECOC-GA ensemble

learning algorithm and the two traditional ensemble learning algorithms is illustrated in Figure 3. It can be found that the classification accuracy of Random Forest and XGBoost is lower, and the ECOC-GA algorithm has a significant improvement in accuracy for performance prediction compared with them, indicating that the ECOC-GA algorithm outperforms traditional ensemble learning algorithms.

5.2.2. Comparison with Other ECOC Algorithms. On the basis of ECOC, ECOC-GA algorithm achieves better prediction performance using the feature selection and base classifier selection through GA. The experimental results between the two kinds of algorithms are summarized in Tables 4–6.

As shown in Table 4, when OVO coding method is adopted in the experiments: The prediction accuracy of the ECOC-GA on the grade 1 dataset is improved by about 7.4%, 6.2%, and 11.7% over ECOC (SVM), ECOC (Bayes), and ECOC (LR), respectively; the prediction accuracy of the ECOC-GA on the grade 2 dataset is increased by about 1.4%, 12.5%, and 1.3% over ECOC (SVM), ECOC (Bayes), and ECOC (LR), respectively; the prediction accuracy of the ECOC-GA on the grade 3 dataset is lifted by about 1.7%, 3.5%, and 0.4% over ECOC (SVM), ECOC (Bayes), and ECOC (LR), respectively. Therefore, the proposed algorithm has significantly improved the prediction accuracy over classical ECOC algorithms on all three datasets.

When DR coding method is used, we can see from Table 5 that the prediction accuracy of the ECOC-GA on the grade 1 dataset is improved by about 0.7%, 3.7%, and 9.9%, respectively, compared with ECOC (SVM), ECOC (Bayes), and ECOC (LR). Compared with ECOC (SVM), ECOC (Bayes), and ECOC (LR), the prediction accuracy of ECOC-GA on grade 2 dataset is increased by about 1.8%, 3.5%, and 1.3%, respectively. Compared with ECOC (SVM), ECOC (Bayes), and ECOC (LR), the prediction accuracy of ECOC-GA on grade 3 dataset is lifted by about 1.7%, 5.9%, and 4.8%, respectively. Thus, using DR coding method, the ECOC-GA also has a small performance improvement in three datasets.

Similar results are also reflected in the experiments using DECOC coding method. As demonstrated in Table 6, when DECOC coding method is applied, the prediction accuracy of ECOC-GA on the grade 1 dataset grows by about 3.9%, 1.5%, and 8.9%, respectively, compared with ECOC (SVM), ECOC (Bayes), and ECOC (LR). Compared with ECOC (SVM), ECOC (Bayes), and ECOC (LR), the prediction accuracy of ECOC-GA on grade 2 dataset is improved by 3.0%, 11.6%, and 3.9%, respectively. Compared with ECOC (SVM), ECOC (Bayes), and ECOC (LR), the prediction accuracy of ECOC-GA on grade 3 dataset is increased by about 0.5%, 5.3%, and 3.1%, respectively.

In addition, we also observe that the algorithms have slightly lower accuracy on the grade 3 dataset. The possible reason is that the grade 3 dataset has more dimensions of features, and the corresponding solution space is larger than the other two datasets. Hence, it is more difficult to search

for an optimized parameter, and the classification accuracy is reduced. However, on the whole, the prediction accuracy of ECOC-GA improved to different degrees over ECOC algorithms on all three datasets.

In summary, even with different datasets or coding methods, the prediction accuracy of the ECOC-GA is consistently higher than that of other ECOC algorithms. Therefore, the ECOC algorithm with GA for feature selection and base classifiers selection has a superior performance.

5.2.3. Comparison of Different Methods at Different Time Stages. Furthermore, in order to find out students' learning effectiveness at different learning stages so that teachers can intervene effectively in advance, we use the ECOC-GA method to predict students' learning performance at different time stages. Subsequently, we compare the performance of algorithms at different learning stages. Specifically, from the end of midterm examination to the end of the course, every two weeks is considered as a learning stage, and a total of six stages can be formed. Experiments are conducted using ECOC, ECOC-GA, Random Forest, and XGBoost, respectively.

For different datasets, different base classifiers, and different coding methods, there are obvious differences in prediction accuracy between algorithms based on ECOC, i.e., ECOC and ECOC-GA, and classical ensemble learning algorithms, i.e., XGBoost and Random Forest. Because the ECOC-GA uses multiple base classifiers, it is not fair to compare it directly with other ECOC algorithms. Therefore, we select the best results among three ECOC algorithms based on single-classifier and compare them with the ECOC-GA.

When OVO is used, the results can be observed from Figure 4. With 18 cases on different datasets, the ECOC-GA algorithm can all obtain better results. Figure 5 shows that there are 18 cases in the datasets at different time stages when DR is used. The ECOC-GA can achieve 17 better results, 15 better results, and 18 better results in the datasets of grades 1, 2, and 3 at different time stages, respectively. Finally, as can be observed from Figure 6, there are also 18 cases on different datasets when DECOC is used. The ECOC-GA wins 15, 15, and 16 cases out of 18 cases on different time stage datasets at grades 1, 2, and 3, respectively.

In general, although the ECOC-GA does not always achieve better results than ECOC algorithms, it generally outperforms ECOC algorithms and has stronger robustness. In addition, XGBoost and Random Forest are less effective in predicting results at different time stages. These results fully demonstrate the good classification performance of the proposed ECOC-GA on the learning behaviors datasets.

5.2.4. Comparison of Time Complexity. Table 7 provides the running time of ECOC-GA. On the three datasets, both Random Forest and XGBoost can complete the training within 20 seconds, thus having the lowest complexity. ECOC algorithms can complete the training within 50 seconds regardless of different coding methods and different base

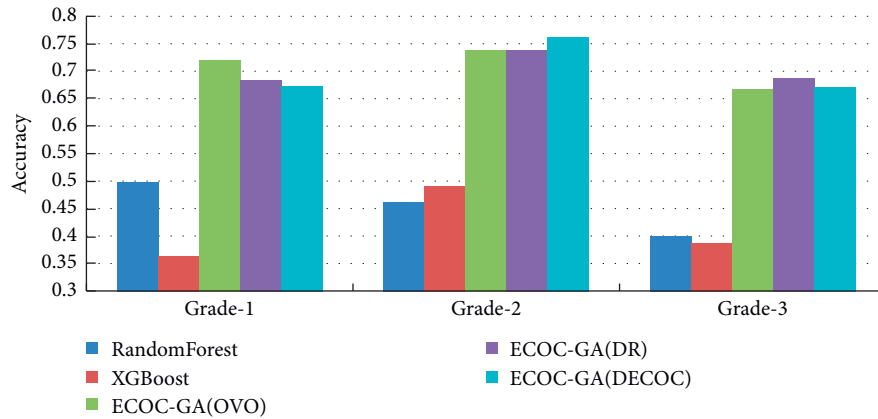


FIGURE 3: Average accuracy of ECOC-GA and traditional ensemble learning algorithms.

TABLE 4: Average accuracy of ECOC-GA and ECOC with OVO coding method.

Datasets	ECOC (SVM)	ECOC (Bayes)	ECOC (LR)	ECOC-GA
Grade 1	0.646 ± 0.043	0.658 ± 0.063	0.603 ± 0.043	0.720 ± 0.082
Grade 2	0.722 ± 0.045	0.611 ± 0.034	0.723 ± 0.037	0.736 ± 0.030
Grade 3	0.648 ± 0.044	0.630 ± 0.048	0.661 ± 0.054	0.665 ± 0.021

TABLE 5: Average accuracy of ECOC-GA and ECOC with DR coding method.

Datasets	ECOC (SVM)	ECOC (Bayes)	ECOC (LR)	ECOC-GA
Grade 1	0.676 ± 0.049	0.646 ± 0.062	0.584 ± 0.059	0.683 ± 0.055
Grade 2	0.717 ± 0.030	0.700 ± 0.058	0.722 ± 0.031	0.735 ± 0.029
Grade 3	0.670 ± 0.039	0.628 ± 0.048	0.639 ± 0.038	0.687 ± 0.029

TABLE 6: Average accuracy of ECOC-GA and ECOC with DECOC coding method.

Datasets	ECOC (SVM)	ECOC (Bayes)	ECOC (LR)	ECOC-GA
Grade 1	0.634 ± 0.043	0.658 ± 0.053	0.584 ± 0.063	0.673 ± 0.049
Grade 2	0.731 ± 0.028	0.645 ± 0.034	0.722 ± 0.034	0.761 ± 0.035
Grade 3	0.665 ± 0.038	0.617 ± 0.040	0.639 ± 0.029	0.670 ± 0.046

classifiers, so the complexity of the algorithms is also low. On the contrary, the ECOC-GA requires more than 3000 seconds to fit models. Hence, the complexity is relatively high.

Because the number of samples and features of the three datasets is different, the running time of the methods are also different. Grade 3 dataset has 290 dimensional features; the solution space available for searching is larger, and the computing time is longer. Grade 1 dataset has 137 features, and grade 2 dataset has 150 features. The computing time of them is similar, and both shorter than that of grade 3 dataset.

On the other hand, for the same dataset, the time complexity of different algorithms is not much different. Nevertheless, the average operation time using DR is slightly larger than OVO, whereas the average operation time using DECOC is the shortest. The reason is that when employing DR coding method to generate random matrix, if there is any illegal coding matrix, it should be deleted and new coding matrix is needed. However, the matrix generated by OVO is fixed as well as simple, and therefore, the complexity of the algorithm is smaller.

5.3. Analysis of the Educational Significance of Key Features. By calculating the Pearson correlation coefficient between students' learning behaviors and students' final grades, the

influence of different features on students' final grades can be observed.

5.3.1. Correlation Analysis of Features. Correlation analysis is performed to identify the features that are significantly correlated with grades. The Pearson correlation coefficient of each feature and final grade can be obtained. For the grade 1 dataset, by filtering the features with high Pearson correlation coefficients, we find high linear correlations between the total scores, attendance rates, cnBlog assignment scores, completion of some courseware, the usual performance, and the final grades for Rain Classroom. For the dataset of grade 2, it is still found that these features have a high linear correlation with final grades through filtering. The proportion of features with high correlation to the total features is high as well, which indicates that students in this grade may have higher utilization of e-learning platform. For the grade 3 dataset, through screening, we find that in addition to the above features, the chapter quizzes and final grades in MOOCs have a high correlation, and the rest of the MOOC features have a low correlation. The dimensions of dataset in grade 3 are high, but there are few features with high correlation with final

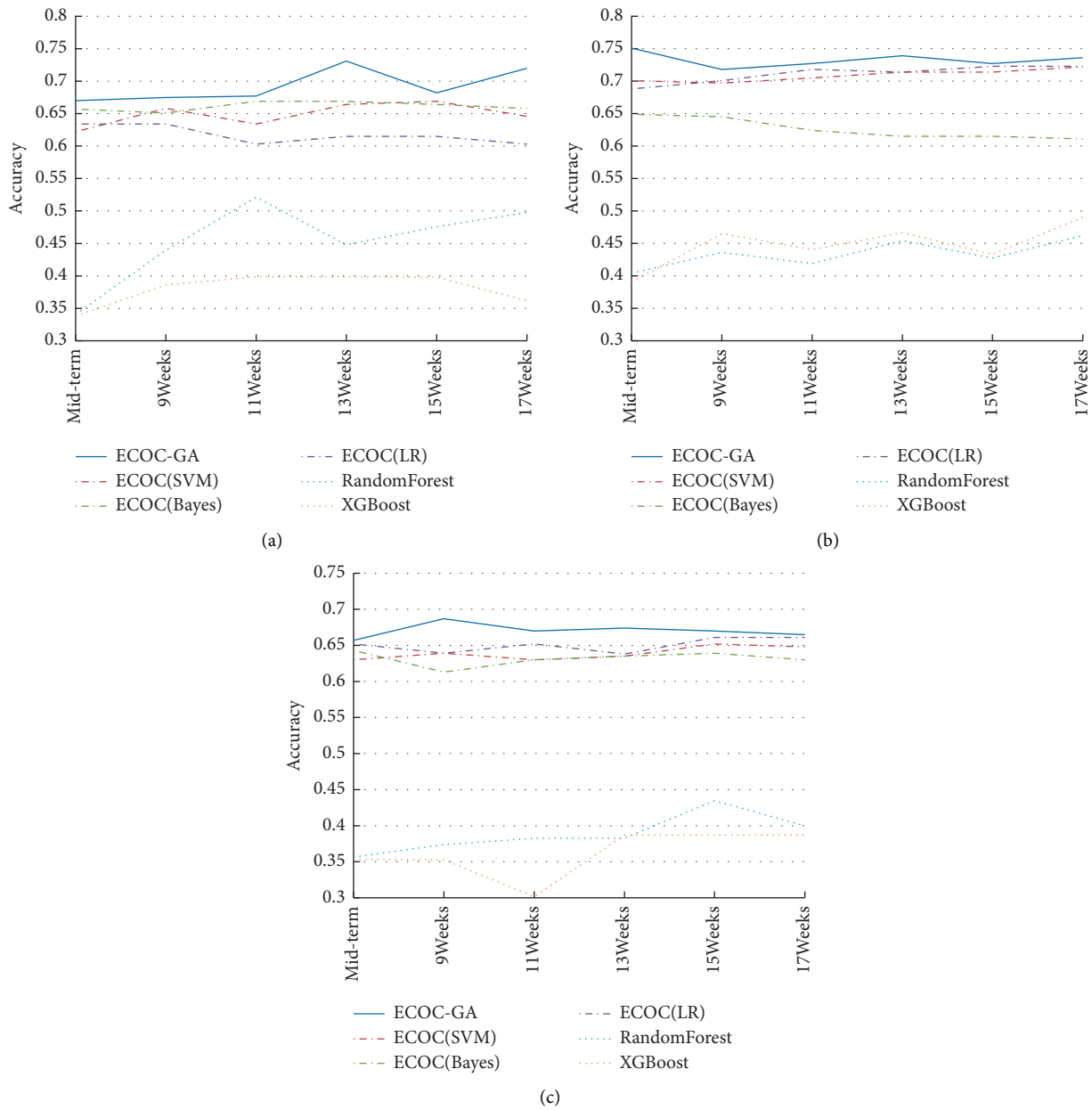


FIGURE 4: Accuracy comparison of different algorithms with OVO coding method. (a) Grade 1 dataset. (b) Grade 2 dataset. (c) Grade 3 dataset.

grades, which indicates that students in this grade may not pay much attention to e-learning, and their overall low final grades may also confirm this conjecture.

5.3.2. *Analysis of the Optimal Feature Subset and Its Educational Implications.* Using GA as the feature selection algorithm, the feature selection result corresponding to the final generation is the most optimal feature subset, and the accuracy of using this feature subset is also the highest. Therefore, these features are likely to have specific pedagogical significance. The ECOC-GA designed in this study has good prediction accuracy, so we decide to utilize the common features selected by the ECOC-GA based on the

three coding methods as the final result. Subsequently, we will analyze these features to extract useful information.

Due to the high dimensionality of the data, the number of retained features is still high after performing feature selection, although nearly half of the features are filtered. Thus, it is decided to combine the results of the correlation analysis and use the intersection of the features selected by the two methods to further cut down the number of features and select the features that have a higher degree of influence on the final grades. It is beneficial to analyze the pedagogical implications of the selected features. Specifically, for grade 1 dataset, 23 of the 42 features with high correlation are also selected by feature selection; for grade 2 dataset, 32 of the 60 features are selected; for grade 3 dataset, 18 of the 32 features

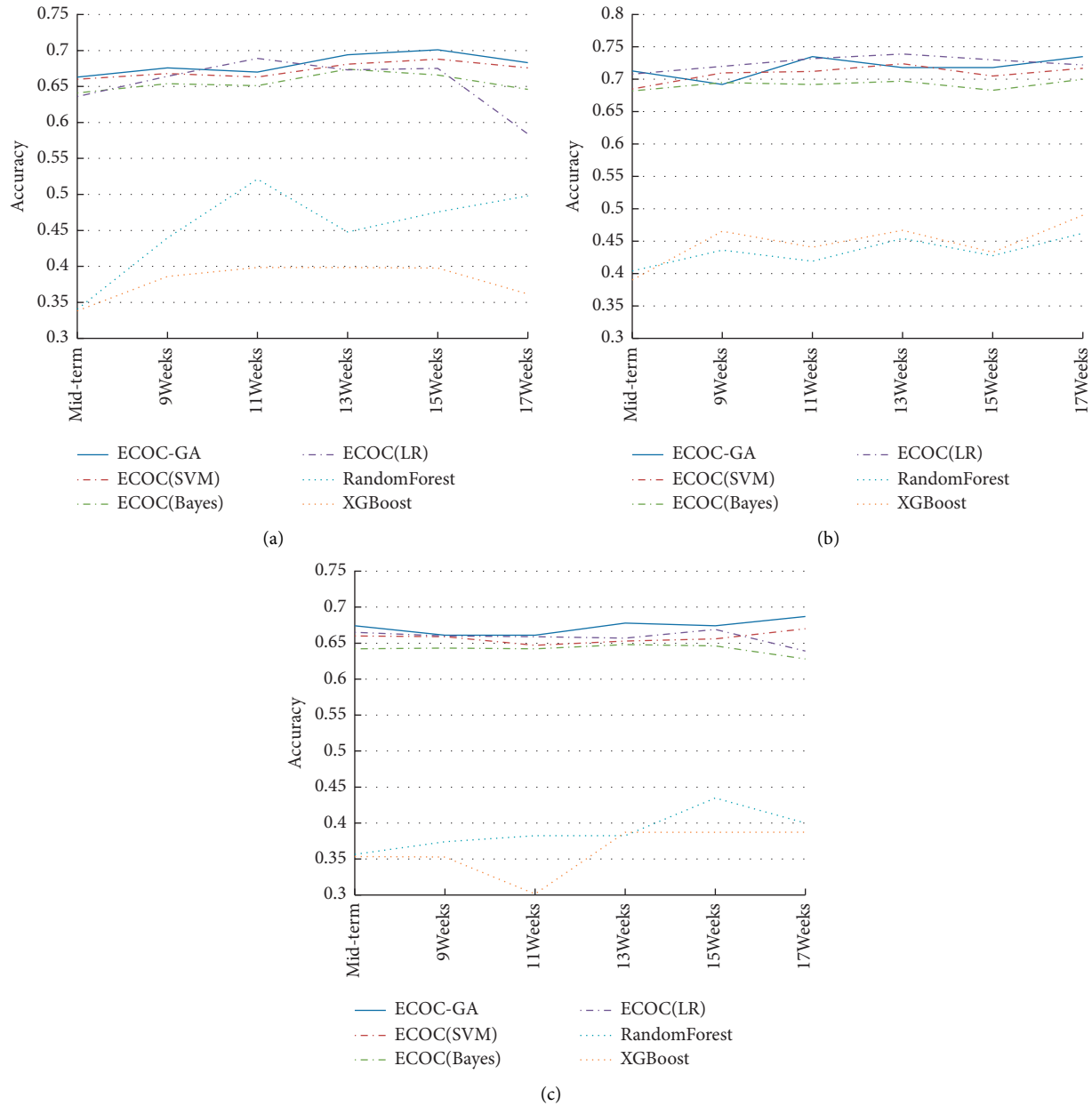


FIGURE 5: Accuracy comparison of different algorithms with DR coding method. (a) Grade 1 dataset. (b) Grade 2 dataset. (c) Grade 3 dataset.

are selected. There are 18 features among the 32 features with high correlation. The results also show that there is a corroborating relationship between the results of the correlation analysis and the results of the feature selection to some extent. The results are displayed in Table 8, where “courseware” indicates students’ prereading of the current content, “class status” indicates students’ learning performance in the current course content, and “xxx.mp4” refers to the progress and number of times students watched the video.

As listed in Table 8, for the students of grade 1, the learning contents of Java Language Programming such as Inheritance, Polymorphism, Interfaces, Nested classes, Collections, Object-Oriented, Exceptions, Files and Databases, and Multithreading have a significant impact on the

final grades. In addition, the completion of students’ cnBlog assignments also has an important impact on the final grades. For the students of grade 2, Object-Oriented, Inheritance, Polymorphism, Interfaces, Nested classes, Collections, Exceptions, Multithreading, and I/O flow have a high correlation with the final grade. For the students of grade 3, the main features that show high correlation are analogous to the aforementioned important features in terms of watching videos and previewing courseware alike. From the intersection of features of the above three datasets, it can be observed that Collections, Object-Oriented, Inheritance, Polymorphism, Interfaces, Nested classes, and Exceptions are the key and difficult points of the whole course. Hence, good command of relevant knowledge contributes greatly to students’ final grades. The correctness

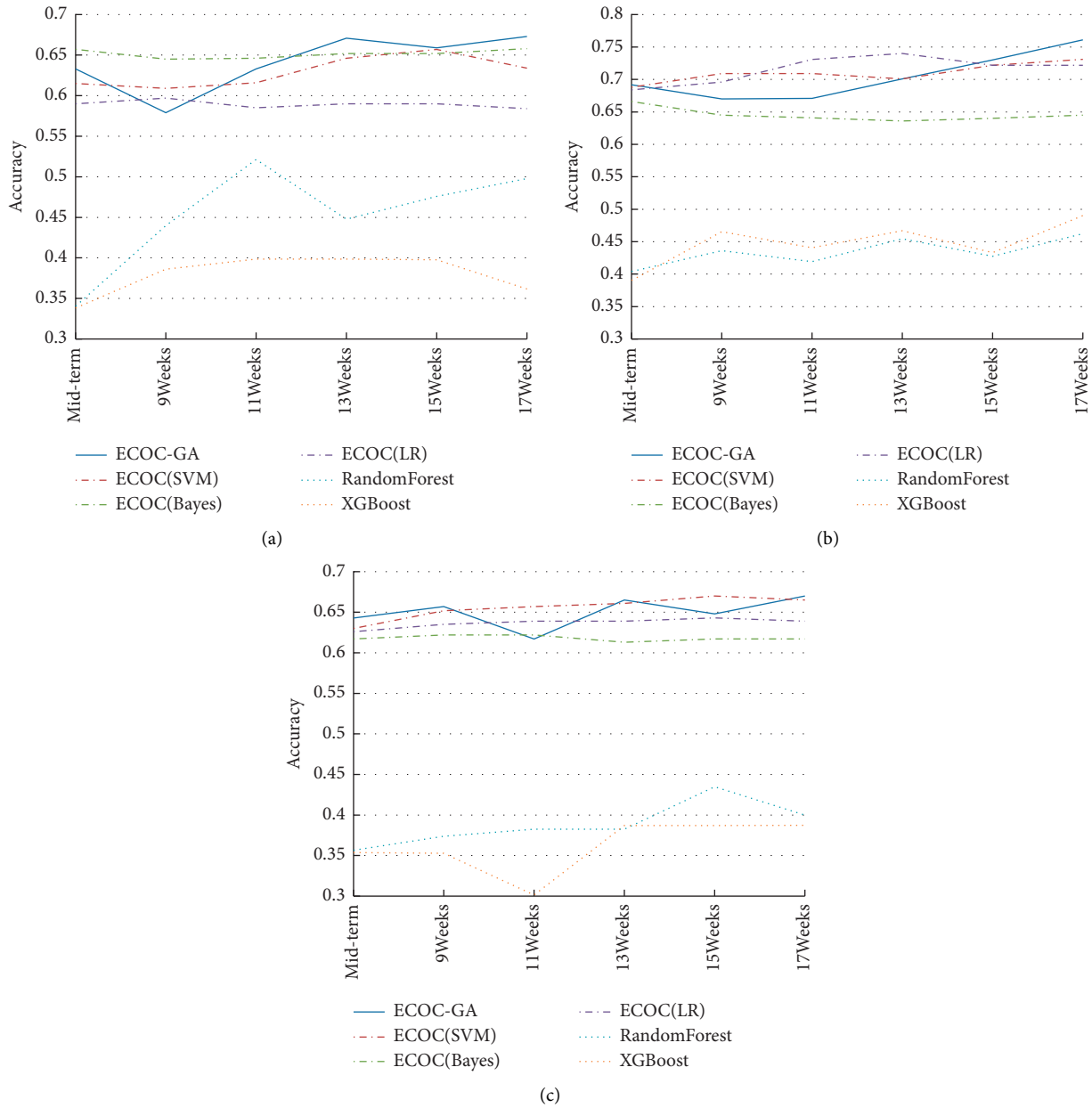


FIGURE 6: Accuracy comparison of different algorithms with DECOC coding method. (a) Grade 1 dataset. (b) Grade 2 dataset. (c) Grade 3 dataset.

TABLE 7: The running time of ECOC-GA (in seconds).

Datasets	ECOC-GA (OVO)	ECOC-GA (DR)	ECOC-GA (DECOC)	ECOC	XGBoost	Random Forest
Grade 1	3250	3332	3158	≈50	≈20	≈20
Grade 2	3544	3608	3392	≈50	≈20	≈20
Grade 3	5535	5808	5472	≈50	≈20	≈20

of the inference is further confirmed by the content selected in several assignments in the cnBlog.

The selected features can be organized into three major categories of learning behaviors: preview before class, course learning, and quizzes after class. To be concrete, the preview before class is divided into PPT viewing and precourse practice; the course learning is divided into videos watching and in-class practice; the quizzes after class is divided into

after-class homework and chapter quizzes. From the experimental results, it is clear that the pages of PPT viewed before class, the scores of in-class quizzes, and the completion of homework are more helpful to improve the final grades. As to the reason for this phenomenon, we believe that the more PPT pages students viewed before class, in other words, the greater his efforts and investment put in the current content, the better learning outcomes they will get in return. Besides,

TABLE 8: Feature selection results.

Datasets	Selected features
Grade 1	<i>Total score</i> , attendance rate, Java data types, and reference types—courseware, object-oriented programming—courseware, <i>inheritance and polymorphism</i> —courseware, interfaces and nested classes—courseware, GUI and swing—courseware, <i>Java collection framework and object oriented</i> —courseware, <i>exceptions</i> —courseware, multithreading—class status, networking—courseware, file and database—courseware, JSP and servlet—courseware, exception handling—class status, cnBlog (2nd, 3rd, 9th, 12th, 13th, and 15th assignment), <i>usual performance</i>
Grade 2	<i>Total score</i> , number of reading announcements, procedural oriented programming—courseware, <i>objects and classes</i> —courseware, <i>inheritance and polymorphism</i> —courseware, <i>interfaces and nested classes</i> —courseware, Java swing Java’s GUI—courseware, <i>Java collection framework</i> —courseware, <i>Java exceptions</i> —courseware, exception mechanism—class status, Java multithreading—courseware, Java input-output flow—courseware, database—class status, servlet and JSP—courseware, PTA midterm examination scores, <i>usual performance</i>
Grade 3	<i>Collections.mp4</i> , <i>Exception_Exception_Handling.mp4</i> , chapter quiz progress, collections—peer assessment, exceptions—chapter quiz, object oriented—chapter quiz, <i>interfaces and nested</i> —chapter quiz, collections—chapter quiz, <i>objects and classes</i> —courseware, exception mechanisms—class status, <i>inheritance and polymorphism</i> —courseware, I/O streams and file operations—class status, Java—network programming—courseware, PTA midterm examination scores, <i>usual performance</i>

the interspersed exercises assist students to verify their mastery of the content during the learning process. By doing this, students can consolidate what they have learned, and their sense of achievement can be stimulated greatly so that more efficient learning can be achieved, which can further create a virtuous circle. Meanwhile, the completion of after-lesson assignments reflects the students’ learning attitudes to some extent, so students who do well in cnBlog assignments must take every aspect of their online course learning seriously, which determines their higher final grades.

Another interesting phenomenon is that very few features of MOOC platform are selected, suggesting that students’ use of MOOC for e-learning does not correlate well with their final grades. Through communication with course instructors, it is found that MOOCs are only used as a teaching aid, so students may not pay much attention to it or have much investment. The viewing of more than 100 videos is also not very relevant to the final grades, probably because students are only “task-oriented” on the course in order to complete the task, which naturally does not reflect the students’ knowledge mastery. In addition, the PTA midterm examination results did not have a close relationship with the final grades, which is also worth considering and needs to be further explored.

In summary, the selected features have significant pedagogical significance and can aid teachers to fully understand the arrangement of teaching content and to suit the remedy to the case in future teaching by modifying the teaching content and curriculum appropriately as well as make full use of blended teaching methods to improve teaching effectiveness.

6. Conclusion

In this article, we propose a novel ECOC based on GA, which is applied to the mining of blended teaching data. The main findings are as follows:

- (1) It is experimentally demonstrated that the ECOC-GA algorithm has a large improvement in accuracy of prediction when compared with classical ECOC algorithms and the traditional ensemble learning algorithm, such as XGBoost and Random Forest. In

addition, the prediction performance of ECOC-GA is also better than ECOC algorithms, XGBoost, and Random Forest algorithm in general for prediction at different periods.

- (2) Through the combinations of feature selection and correlation analysis, the selected feature subsets are analyzed for their pedagogical significance, and the key and difficult points of the “*Java Language Programming*” course are found. On the basis of them, teachers can improve the teaching setup, enhance teaching quality, and help students consolidate key points and break through difficult points, so that students can effectively improve their performance. At the same time, we find that the cnBlog assignments are essentially helpful, which can be promoted and further enriched in future teaching. Conversely, MOOCs and PTA are not effective enough in practice. Accordingly, teachers should have second thoughts about the use and setting of MOOCs and PTA.

Future work can be improved in the following ways. Firstly, improve data quality and collect more teaching-related data. Secondly, improve the ECOC algorithm to make it more suitable for fitting smaller samples of higher dimensional data, thus further improving the prediction accuracy. Finally, GA can not only be used for feature selection but also can be combined with ECOC algorithm. So future research can explore the better ways of their combination.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the Natural Science Foundation of Fujian Province of China (nos. 2020J01697,

2020J01707, 2020R0066, and 2018J01538), the Scientific Research Program of Fujian Bureau of Education, China (no. JAT200266), and the Opening Fund of Digital Fujian Big Data Modeling and Intelligent Computing Institute.

References

- [1] K. L.-M. Ang, F. L. Ge, and K. P. Seng, "Big educational data & analytics: survey, architecture and challenges," *IEEE Access*, vol. 8, pp. 116392–116414, 2020.
- [2] J. Qiu, J. Tang, T. X. Liu et al., "Modeling and predicting learning behavior in MOOCs," in *WSDM '16: Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*, pp. 93–102, San Francisco, CA, USA, February 2016.
- [3] B. Xu and D. Yang, "Motivation classification and grade prediction for MOOCs learners," *Computational Intelligence and Neuroscience*, vol. 2016, Article ID 2174613, 7 pages, 2016.
- [4] M. Zhang, J. Zhu, Z. Wang, and Y. Chen, "Providing personalized learning guidance in MOOCs by multi-source data analysis," *World Wide Web*, vol. 22, no. 3, pp. 1189–1219, 2019.
- [5] C.-H. Yu, J. Wu, and A.-C. Liu, "Predicting learning outcomes with MOOC clickstreams," *Education Sciences*, vol. 9, no. 2, p. 104, 2019.
- [6] Y. Meier, J. Xu, O. Atan, and M. Van der Schaar, "Predicting grades," *IEEE Transactions on Signal Processing*, vol. 64, no. 4, pp. 959–972, 2015.
- [7] R. L. Ulloa-Cazarez, C. López-Martín, A. Abran, and C. Yáñez-Márquez, "Prediction of online students performance by means of genetic programming," *Applied Artificial Intelligence*, vol. 32, no. 9-10, pp. 858–881, 2018.
- [8] J. Xu, K. H. Moon, and M. Van Der Schaar, "A machine learning approach for tracking and predicting student performance in degree programs," *IEEE Journal of Selected Topics in Signal Processing*, vol. 11, no. 5, pp. 742–753, 2017.
- [9] M. Hussain, W. Zhu, W. Zhang, S. M. R. Abidi, and S. Ali, "Using machine learning to predict student difficulties from learning session data," *Artificial Intelligence Review*, vol. 52, no. 1, pp. 381–407, 2018.
- [10] H. Hu, G. Zhang, W. Gao, and M. Wang, "Big data analytics for MOOC video watching behavior based on Spark," *Neural Computing & Applications*, vol. 32, no. 11, pp. 6481–6489, 2019.
- [11] K. F. Hew, X. Hu, C. Qiao, and Y. Tang, "What predicts student satisfaction with MOOCs: a gradient boosting trees supervised machine learning and sentiment analysis approach," *Computers & Education*, vol. 145, Article ID 103724, 2020.
- [12] A. Onan and M. A. Toçoğlu, "Weighted word embeddings and clustering-based identification of question topics in MOOC discussion forum posts," *Computer Applications in Engineering Education*, vol. 29, no. 4, pp. 675–689, 2020.
- [13] A. Onan, "sentiment analysis on massive open online course evaluations: A text mining and deep learning approach," *Computer Applications in Engineering Education*, vol. 29, no. 3, pp. 572–589, 2020.
- [14] O. Aytuğ, "Sentiment analysis on Twitter based on ensemble of psychological and linguistic feature sets," *Balkan Journal of Electrical and Computer Engineering*, vol. 6, no. 2, pp. 69–77, 2018.
- [15] A. Onan, "Mining opinions from instructor evaluation reviews: a deep learning approach," *Computer Applications in Engineering Education*, vol. 28, no. 1, pp. 117–138, 2020.
- [16] A. Onan, "Sentiment analysis on product reviews based on weighted word embeddings and deep neural networks," *Concurrency and Computation: Practice and Experience*, Article ID e5909, 2020.
- [17] X. Chen, J. Zheng, Y. Du, and M. Tang, "Intelligent course plan recommendation for higher education: a framework of decision tree," *Discrete Dynamics in Nature and Society*, vol. 2020, pp. 1–11, 2020.
- [18] A. M. F. Yousef and T. Sumner, "reflections on the last decade of MOOC research," *Computer Applications in Engineering Education*, vol. 29, no. 4, pp. 648–665, 2020.
- [19] A. A. Mubarak, H. Cao, W. Zhang, and W. Zhang, "Visual analytics of video-clickstream data and prediction of learners' performance using deep learning models in MOOCs' courses," *Computer Applications in Engineering Education*, vol. 29, no. 4, pp. 710–732, 2020.
- [20] L. Zhang, Y. Xuan, and H. Zhang, "Construction and application of SPOC-based flipped classroom teaching mode in Installation Engineering Costcurriculum based on OBE concept," *Computer Applications in Engineering Education*, vol. 28, no. 6, pp. 1503–1519, 2020.
- [21] Q. Chen, X. Yue, X. Plantaz et al., "Visual analytics of learning sequence in massive open online courses," *IEEE Transactions on Visualization and Computer Graphics*, vol. 26, no. 3, pp. 1622–1636, 2018.
- [22] Y. Wang, C. Dong, and X. Zhang, "Improving MOOC learning performance in China: an analysis of factors from the TAM and TPB," *Computer Applications in Engineering Education*, vol. 28, no. 6, pp. 1421–1433, 2020.
- [23] R. Cobos and J. C. Ruiz-García, "Improving learner engagement in MOOCs using a learning intervention system: A research study in engineering education," *Computer Applications in Engineering Education*, vol. 29, no. 4, pp. 733–749, 2020.
- [24] S. García-Molina, C. Alario-Hoyos, P. M. Moreno-Marcos, P. J. Muñoz-Merino, I. Estévez-Ayres, and C. Delgado Kloos, "An algorithm and a tool for the automatic grading of MOOC learners from their contributions in the discussion forum," *Applied Sciences*, vol. 11, no. 1, p. 95, 2021.
- [25] Z. Xie, "Modelling the dropout patterns of MOOC learners," *Tsinghua Science and Technology*, vol. 25, no. 3, pp. 313–324, 2020.
- [26] Y. Wen, Y. Tian, B. Wen, Q. Zhou, G. Cai, and S. Liu, "Consideration of the local correlation of learning behaviors to predict dropouts from MOOCs," *Tsinghua Science and Technology*, vol. 25, no. 3, pp. 336–347, 2020.
- [27] C.-H. Yu, J. Wu, M.-C. Liu, and A.-C. Liu, "Adopting software product lines to implement an efficient learning analytics framework in MOOCs," *Journal of Information Science and Engineering*, vol. 37, no. 1, pp. 139–155, 2021.
- [28] M. Sun, K. Liu, Q. Wu, Q. Hong, B. Wang, and H. Zhang, "A novel ECOC algorithm for multiclass microarray data classification based on data complexity analysis," *Pattern Recognition*, vol. 90, pp. 346–362, 2019.
- [29] X.-N. Ye, K.-H. Liu, and S.-T. Liong, "A ternary bitwise calculator based genetic algorithm for improving error correcting output codes," *Information Sciences*, vol. 537, pp. 485–510, 2020.
- [30] Y.-P. Zhang, X.-N. Ye, K.-H. Liu, and J.-F. Yao, "A novel multi-objective genetic algorithm based error correcting output codes," *Swarm and Evolutionary Computation*, vol. 57, Article ID 100709, 2020.

- [31] T. G. Dietterich and G. Bakiri, "Solving multiclass learning problems via error-correcting output codes," *Journal of Artificial Intelligence Research*, vol. 2, pp. 263–286, 1994.
- [32] M. Galar, A. Fernández, E. Barrenechea, H. Bustince, and F. Herrera, "NMC: nearest matrix classification - a new combination model for pruning One-vs-One ensembles by transforming the aggregation problem," *Information Fusion*, vol. 36, pp. 26–51, 2017.
- [33] K. Crammer and Y. Singer, "On the learnability and design of output codes for multiclass problems," *Machine Learning*, vol. 47, no. 2-3, pp. 201–233, 2002.
- [34] L. Zhou, Q. Wang, and H. Fujita, "One versus one multi-class classification fusion using optimizing decision directed acyclic graph for predicting listing status of companies," *Information Fusion*, vol. 36, pp. 80–89, 2017.
- [35] F. Masulli and G. Valentini, "Effectiveness of error correcting output coding methods in ensemble and monolithic learning machines," *Formal Pattern Analysis & Applications*, vol. 6, no. 4, pp. 285–300, 2004.
- [36] D. Dutta, J. Sil, and P. Dutta, "Automatic clustering by multi-objective genetic algorithm with numeric and categorical features," *Expert Systems with Applications*, vol. 137, pp. 357–379, 2019.
- [37] M. A. Bagheri, Q. Gao, and S. Escalera, "A genetic-based subspace analysis method for improving Error-Correcting Output Coding," *Pattern Recognition*, vol. 46, no. 10, pp. 2830–2839, 2013.
- [38] L. I. Kuncheva and L. C. Jain, "Designing classifier fusion systems by genetic algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 4, no. 4, pp. 327–336, 2000.

Research Article

The Investigation of Different Loss Functions with Capsule Networks for Speech Emotion Recognition

Anfernee Joan B. Ng  and Kun-Hong Liu 

School of Informatics, Xiamen University, Xiamen, Fujian, China

Correspondence should be addressed to Kun-Hong Liu; lkhqz@xmu.edu.cn

Received 30 May 2021; Revised 1 August 2021; Accepted 8 August 2021; Published 18 August 2021

Academic Editor: Antonio J. Peña

Copyright © 2021 Anfernee Joan B. Ng and Kun-Hong Liu. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Speech emotion recognition (SER) is an important research topic. Image features like spectrograms are one of the common ways of extracting information from speech. In the area of image recognition, a relatively novel type of network called capsule networks has shown good and promising results. This study aims to use capsule networks to encode spatial information from spectrograms and analyse its performance when paired with different loss functions. Experiments comparing the capsule network with models from previous works show that the capsule network performs better than them.

1. Introduction

The research field of speech emotion recognition (SER) has a wide range of applications that benefit areas such as human-computer interaction, customer service, and computer games [1]. The general motivation is to identify the emotional state to provide a more personalized and often better user experience. For example, customer service systems can use SER to determine whether a customer is angry or dissatisfied with the aid of their voice throughout the call [2].

In recent years, deep learning is a common framework that has been used in a variety of fields, including SER [3]. One main benefit of using deep learning models is their innate ability to learn new features from a given set of data. Convolutional neural networks (CNNs) are typically used as the basic framework, resulting in many improvements and variations for the CNN in SER [4, 5]. Similarly, recurrent neural networks (RNNs) take advantage of the time dimension in speech and can extract better features that consider temporal relationships between points in a speech sample. Among RNNs, variations like long short-term memory (LSTM) networks and gated recurrent units (GRUs) are also widely used as the main framework in SER research [6, 7].

Another deep learning framework that has been on the trend recently is the capsule network [8]. Its conception mainly addresses the shortcomings of CNNs, including their insensitivity to changes in orientation like rotation and translation. Capsule networks achieve this by using a structure composed of a group of neurons called a capsule. Rather than receiving scalar values from individual neurons on traditional deep neural networks (DNNs), output values are instead vectors whose length and direction describe the pose, orientation, and probability of the existence of the entity being predicted or classified. Like traditional DNNs, the capsule network can be divided into different levels or layers of capsules. The first layer usually handles primitive or roughly simple entities like lines, and further layers manage more complex objects like lines joining together to make an object. Low-level capsules would pass their vector outputs to higher-level capsules, which tend to agree or complement with their outputs. The agreement is analogous to a simple table composed of its individual parts like the legs and surface. The individual parts are situated on a lower layer (legs and surface), which look for capsules in a higher layer (whole table) that “agree” with them. This agreement is determined by applying dynamic routing or routing-by-agreement.

TABLE 1: Loss functions analysed in this paper. y is the true label encoded in one-hot form, \hat{y} is the true label in $+1/-1$ encoding, $\sigma(\cdot)$ denotes probability estimate.

Name	Formula
L1 loss	$\ y - q\ _1$
L2 loss	$\ y - q\ _2^2$
Chebyshev loss	$\max_k \sigma(q)_k - y_k $
Hinge loss	$\sum_k \max(0, (1/2) - \hat{y}_k q_k)$
Squared hinge loss	$\sum_k \max(0, (1/2) - \hat{y}_k q_k)^2$
Cubed hinge loss	$\sum_k \max(0, (1/2) - \hat{y}_k q_k)^3$
Tanimoto loss	$\sum_k \hat{y}_k \sigma(q)_k / \ \hat{y}_k\ _2 + \ \sigma(q)_k\ _2^2 - \sum_k \hat{y}_k \sigma(q)_k$
Cauchy-Schwarz loss	$-\log(\sum_k \hat{y}_k \sigma(q)_k / \ \hat{y}_k\ _2 \ \sigma(q)_k\ _2)$

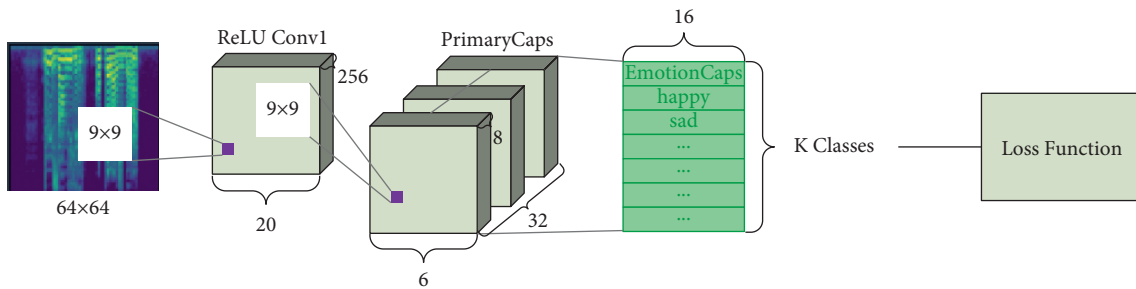


FIGURE 1: Capsule network architecture.

One important consideration in performing deep learning or machine learning in general is the choice of the loss function. Most of the capsule network implementations in other literature [9–11] use the original margin loss as described by Sabour et al. [8]. Only a few have attempted deviating from the original implementation and instead have employed other loss functions. Previous studies [12, 13] have designed custom loss functions but for a specific area or field. To the best of the authors’ knowledge, no other existing literature has reported on the effect of different loss functions used in conjunction with a capsule network. Atmaja and Akagi [14] have published research papers on the analysis of loss functions in the field of SER, but they have not covered them with capsule networks. It is sufficient to say that the impacts of various loss functions on a capsule network are not well understood. If the effects of these loss functions are better understood, then the construction and design of future capsule networks will be more well-informed and easier. In addition, when the choice of a loss function is made easier, researchers can focus on other aspects of their deep learning capsule framework, thereby speeding up their research. As such, the main contribution of this paper is to explore the impacts of different loss functions with the use of a capsule network. Furthermore, this paper also provides insights on the usefulness of these loss functions on multiple SER data sets.

This paper aims to provide an experimental analysis of applying other kinds of loss functions to a capsule network. In a sense, this extends the work done by Janocha and Czarnecki [15], using some of the loss functions experimented there and applying them to a capsule network. The data sets in this paper also differ from the original literature; all of them are taken from the field of SER. In addition, a few baseline models from other papers are tested and compared

with the capsule network. Results show that the capsule network architecture performs slightly better than these baselines.

The remaining contents of this paper are organized as follows. Chapter 2 lays the foundation and theoretical bases needed to understand the model and loss functions analysed in this paper. The same chapter also mentions and explores relevant literature. Chapter 3 explains the methods used in the experiments along with the data sets used. Finally, Chapter 4 provides results and discussion of the said experiments.

2. Relevant Theoretical Bases and Literature

2.1. Recent Advancements. Different techniques in SER classification have been constantly developed and improved over the years. Some have extracted novel types of features like adaptive time-frequency features [16] based on the fractional Fourier transformation and frequency modulation features [17] based on the amplitude modulation-frequency modulation model. In contrast to designing new kinds of features, Özseven [18] instead proposes a novel feature-selection method. The new method involves using multiple statistical measures that are then filtered through a threshold calculated from standard deviations and means between emotional classes.

Aside from features, several previous studies also made improvements on common deep learning models used in SER, such as CNNs and LSTMs. For instance, an ensemble combining DNNs, CNNs, and RNNs was used by Yao et al. [19] to provide different types of features. A confidence-based fusion strategy was also proposed to combine the outputs of these networks in classification. Zhao et al. [20] used different dimensions of CNNs to extract features of

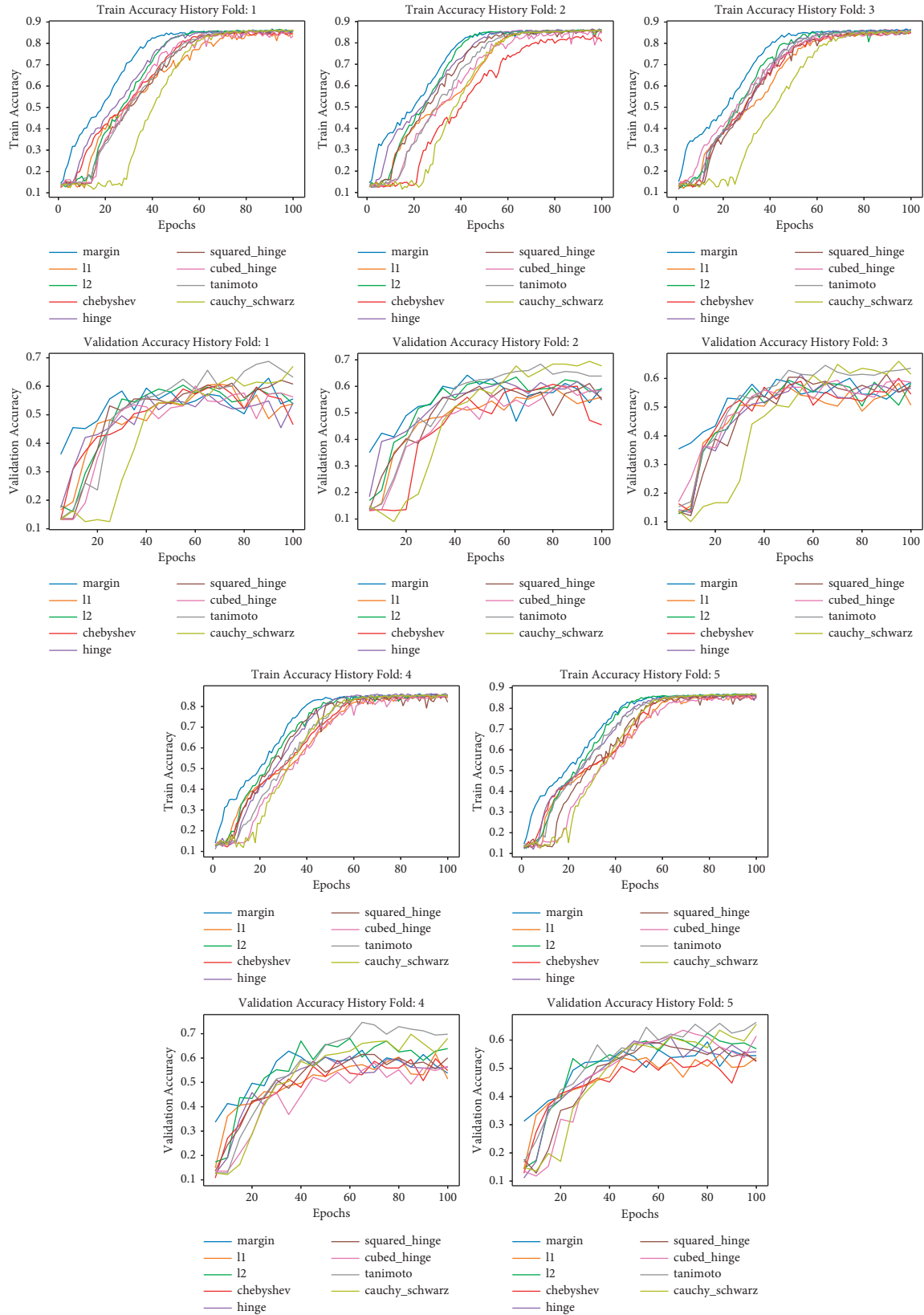


FIGURE 2: RAVDESS train and validation accuracy history for capsule model and different loss functions.

TABLE 2: RAVDESS class F1 scores and accuracies.

Loss type	Neutral (F1)	Calm (F1)	Happy (F1)	Sad (F1)	Angry (F1)	Fearful (F1)	Disgust (F1)	Surprise (F1)	Overall (F1)	Overall (Acc.)
Margin loss	51.83	70.68	54.09	44.38	69.97	67.07	71.40	74.87	63.04	64.38
L1 loss	55.54	66.40	49.92	39.41	67.54	57.46	63.96	67.23	58.43	58.26
L2 loss	58.93	67.03	59.04	46.36	73.16	65.32	64.47	73.04	63.42	64.31
Chebyshev loss	53.36	67.87	49.43	36.30	68.89	56.89	64.52	68.27	58.19	58.89
Hinge loss	53.97	67.67	49.30	39.18	68.50	61.52	62.51	67.91	58.82	59.37
Square hinge loss	51.76	69.78	55.98	49.58	75.56	67.70	69.62	67.91	63.48	64.51
Cubed hinge loss	52.40	73.83	53.09	46.35	69.67	64.16	66.85	70.12	62.06	63.40
Tanimoto loss	66.55	76.75	59.72	55.09	77.12	69.41	73.67	69.05	68.42	69.03
Cauchy-Schwarz loss	62.25	77.72	58.19	57.49	75.80	67.78	75.24	71.91	68.30	68.96

varying granularity, which are then passed to an LSTM network. The role of the LSTM network is to learn global contextual information from the CNN’s resulting features. The researchers discovered that the 2D CNN LSTM network performed better.

2.2. Capsule Network. The basic unit for computation in a capsule network is the namesake itself—“capsule,” which is simply a group of neurons. Unlike regular neurons, capsules output vectors whose length and direction can describe an entity or an object. The length of the vector would represent the probability of the object’s existence in the scene, while the direction or instantiation parameters would provide information on the position, orientation, size, and other properties. A typical network comprises few layers of capsules, with each layer responsible for checking objects of different size or complexity. The first layer is tasked to check for simple or small objects, while the subsequent layers build upon the existence of these primitive objects to compose larger ones. Higher-level capsules do this by receiving activations from lower-level capsules, which are so-called “components” of the more complex object it is trying to predict.

The network determines these lower-to-higher-level capsule relationships using an iterative dynamic routing mechanism. In a nutshell, the dot product is calculated from the “prediction vectors” taken from the previous and the output vector of the current layer and then used to update coupling coefficients which can either strengthen or weaken the relationship between a capsule in the preceding and current layer. In mathematical terms, it can be formulated as

$$\begin{aligned} \mathbf{o}_j &= \sum_i c_{ij} \hat{\mathbf{u}}_{j|i}, \\ \hat{\mathbf{u}}_{j|i} &= \mathbf{W}_{ij} \mathbf{u}_i, \end{aligned} \quad (1)$$

where c_{ij} are the coupling coefficients which are updated at each routing iteration, $\hat{\mathbf{u}}_{j|i}$ is the prediction vector of the previous layer produced by multiplying weight matrix \mathbf{W}_{ij} and output vector \mathbf{u}_i of the previous layer, and \mathbf{o}_j is the preactivation vector for the next layer. This activation function is the squash which ensures that \mathbf{o}_j shrinks to a vector with a length from 0 to 1. The function also has the effect of producing vectors with length close to 0 for short vectors while producing vectors with length close to 1 for long vectors.

$$\mathbf{v}_j = \text{squash}(\mathbf{o}_j) = \frac{\|\mathbf{o}_j\|^2}{1 + \|\mathbf{o}_j\|^2} \frac{\mathbf{o}_j}{\|\mathbf{o}_j\|}. \quad (2)$$

Furthermore, the coupling coefficients c_{ij} are calculated from initial logits b_{ij} which are the log prior probabilities that capsule i should be paired with capsule j . The calculations are designed in such a way that c_{ij} from one specific capsule i all sum up to unity, termed “routing softmax”:

$$c_{ij} = \frac{\exp(b_{ij})}{\sum_k \exp(b_{ik})}. \quad (3)$$

Finally, b_{ij} is updated (thereby updating c_{ij} as well) by adding the scalar product $\mathbf{v}_j \cdot \hat{\mathbf{u}}_{j|i}$, which represents the agreement measure of capsule i and capsule j . Along with \mathbf{W}_{ij} , this process dictates the network’s learning through every iteration. The output vectors \mathbf{v}_k , $1 \leq k \leq K$ (where K is the number of classes) from the last layer will have their magnitudes calculated, afterwards the highest length vector would correspond to the predicted class.

The loss function to be used as a baseline in this paper is from Sabour et al.’s study [8]—the margin loss function:

$$L_k = T_k \max(0, m^+ - \|\mathbf{v}_k\|)^2 + \lambda (1 - T_k) \max(0, \|\mathbf{v}_k\| - m^-)^2, \quad (4)$$

where $T_k = 1$ if the corresponding class k is present, $m^+ = 0.9$, $m^- = 0.1$, and the down-weighting parameter λ for the absent class is 0.5. In addition, L_k will be added onto a reconstruction loss scaled by a factor of 0.0005.

Within the past few years, other studies in the field of speech processing have incorporated the use of capsule-inspired networks. For instance, Lee et al. [21] made use of a CapsNet-only architecture for a sequence-to-sequence speech recognition task. The input sequence was sliced into windows then classified through the same dynamic routing mechanism. The margin loss was replaced by the computation of connectionist temporal classification (CTC). In another paper, Poncelet et al. [10] used capsule networks with recurrent neural networks, additionally encoding time information—an essential property present in speech. They applied this approach in the field of spoken language understanding (SLU). The main focus of this paper, speech emotion recognition, has also received some developments

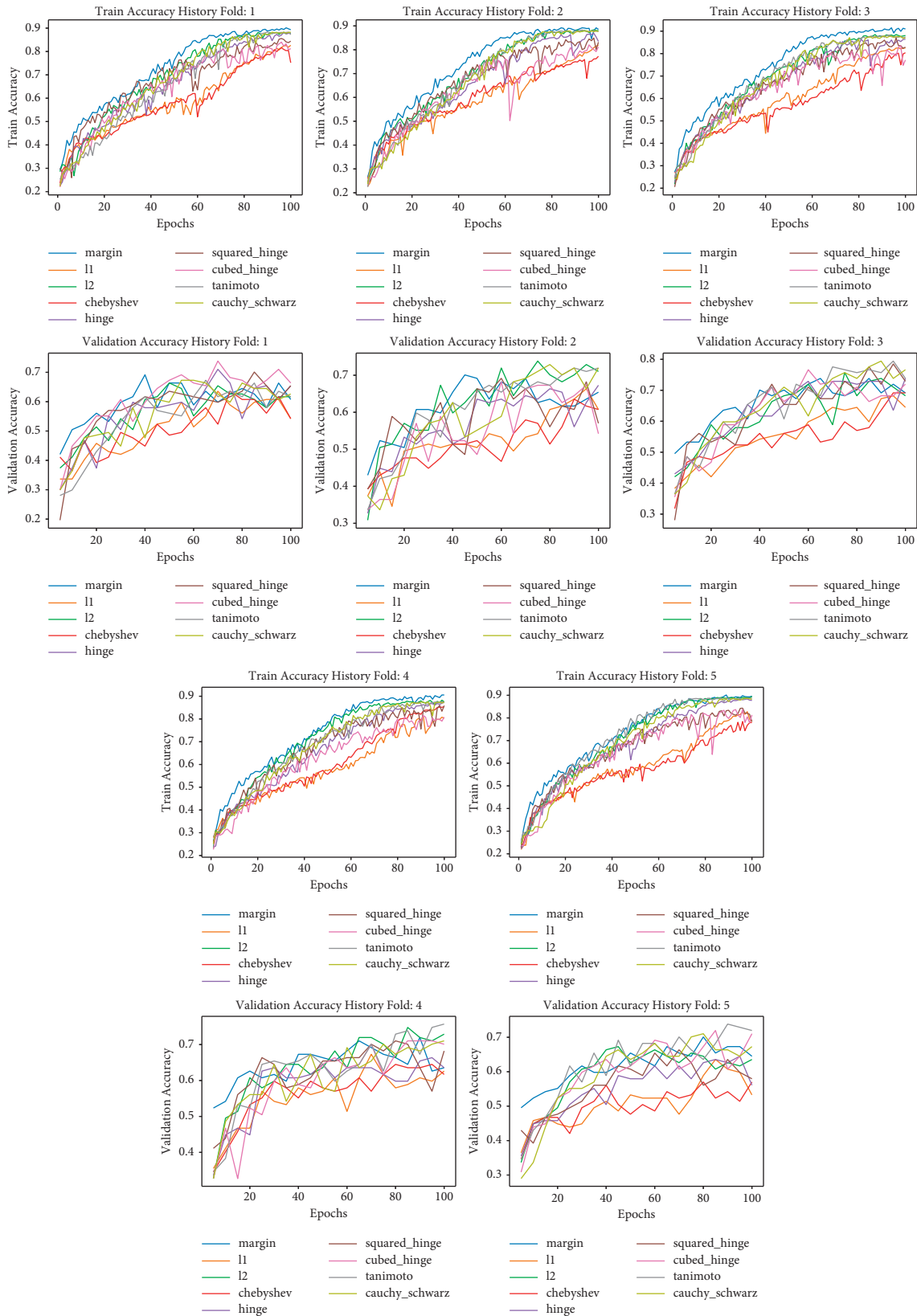


FIGURE 3: EMODB train and validation accuracy history for capsule model and different loss functions.

TABLE 3: EMODB class F1 scores and accuracies.

Loss type	Anger (F1)	Boredom (F1)	Disgust (F1)	Fear (F1)	Happiness (F1)	Sadness (F1)	Neutral (F1)	Overall (F1)	Overall (Acc.)
Margin loss	83.85	72.49	61.04	71.28	58.08	84.32	67.40	71.21	73.46
L1 loss	76.80	67.11	51.92	69.34	39.66	74.50	64.41	63.39	67.10
L2 loss	81.58	69.67	58.68	71.61	54.58	81.11	67.07	69.18	71.59
Chebyshev loss	74.74	65.47	45.73	64.80	34.40	68.74	60.60	59.21	64.30
Hinge loss	81.23	65.45	52.78	64.31	55.55	73.64	64.45	65.35	68.04
Square hinge loss	79.75	66.35	58.90	66.56	57.79	75.01	62.23	66.66	68.60
Cubed hinge loss	78.20	68.00	55.66	70.64	57.23	73.90	64.19	66.83	68.78
Tanimoto loss	81.09	64.60	58.19	67.44	59.22	79.94	64.35	67.83	69.91
Cauchy-Schwarz loss	82.48	66.51	57.09	69.00	57.71	80.98	69.05	68.97	71.61

with the use of capsule networks. These researches mainly use time-frequency spectrograms as their features. Wu et al. [22] improved the capsule network’s performance by adding recurrent connections that can provide the network better feature modelling in the temporal dimension. Wu et al. [22] instead opted for MFCC features as the input for their capsule-based architecture. The capsule network used in this paper is identical to the one proposed by Sabour et al. [8]. Wu et al. [22] and Jain [23] also chose this configuration as well; however, they have added some modifications such as LSTMs and GRUs, further bolstering the feature extraction for the capsule network. This paper instead focuses on the impact of loss functions with the use of a capsule network.

2.3. Loss Functions. The loss functions to be compared in conjunction with the capsule network are listed in Table 1. Also worth noting is that output vectors \mathbf{v}_k have to go through an extra step in order to be more suitable for these loss functions. The output q is calculated from equation (5).

$$q_k = \frac{\|\mathbf{v}_k\|}{\sum_n \|\mathbf{v}_n\|}. \quad (5)$$

L1 and L2 losses are primarily used in regression tasks. Both of these losses are used to complement the primary loss in other classification tasks as a form of regularization. Theoretically speaking, L1 loss is less sensitive to outliers than L2 loss.

The Chebyshev loss is characterized by taking the maximum absolute distance of one of the components between two vectors. Using Chebyshev loss this way would mean that in some cases, even if the model correctly classifies a sample, it may still be heavily penalized if even one component dramatically differs.

Also known as “maximum-margin” loss, hinge loss attempts to maximize the decision boundary between the groups being discriminated against. This type of loss has its origins in support vector machines (SVMs). The squared and cubed variants make the graph smoother and overgrow when the loss gets too big while making errors closer to zero weigh less on optimization.

Tanimoto and Cauchy-Schwarz divergence losses are relatively rarely used in deep learning tasks. The former is similar to Jaccard distance. It measures dissimilarity between

two sampled sets by taking the ratio of the intersection over union among the individual values in the compared vectors. The latter also measures the distance between two random vectors and is an approximation to the Kullback-Leibler divergence [15].

3. Experimental Setup

Four data sets were used to perform the comparison experiments. The first data set is the Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS) [24]. Only the 1,440 speech samples were used in this experiment, spanning across eight emotional classes: calm, happy, sad, angry, fearful, surprise, disgust, and neutral expressions. Each class is equally represented in the database except for the neutral class, which has 96 samples. The rest of the classes each have 192 samples. The database consists of 24 professional actors speaking in a neutral North American accent.

The second data set is the Berlin Emotional Database (EMODB) [25]. It has 535 utterances produced by ten actors (five female and five male) across seven different emotions: neutral, anger, fear, joy, sadness, disgust, and boredom. This data set is quite imbalanced as the difference between the number of samples of the largest and smallest classes is 81, which is alarmingly large for a small data set. The largest class is anger, while disgust was the smallest class.

The third data set is the Canadian French Emotional (CAFE) [26] speech data set with 936 utterances. The data set contains six different sentences, pronounced by 12 actors between two genders. Six basic emotions plus one neutral emotion are represented in the data set. Each class is equally represented except for the neutral emotion, half of one of the other emotions in the data set. The represented emotions are anger, disgust, happiness, fear, surprise, sadness, and a neutral state.

The last data set is the Sharif Emotional Speech Database (SHEMO) [27]. It contains 3000 Persian seminatural utterances extracted from online radio plays. Five emotions plus an extra neutral emotion are included in the data set. These emotions are anger, fear, happiness, sadness, surprise, and a neutral state. Similar to the second data set, a significant difference divides majority and minority classes of around 1000 samples. Anger and neutral emotions have over 1000 samples, while the other emotions have a few hundred samples.

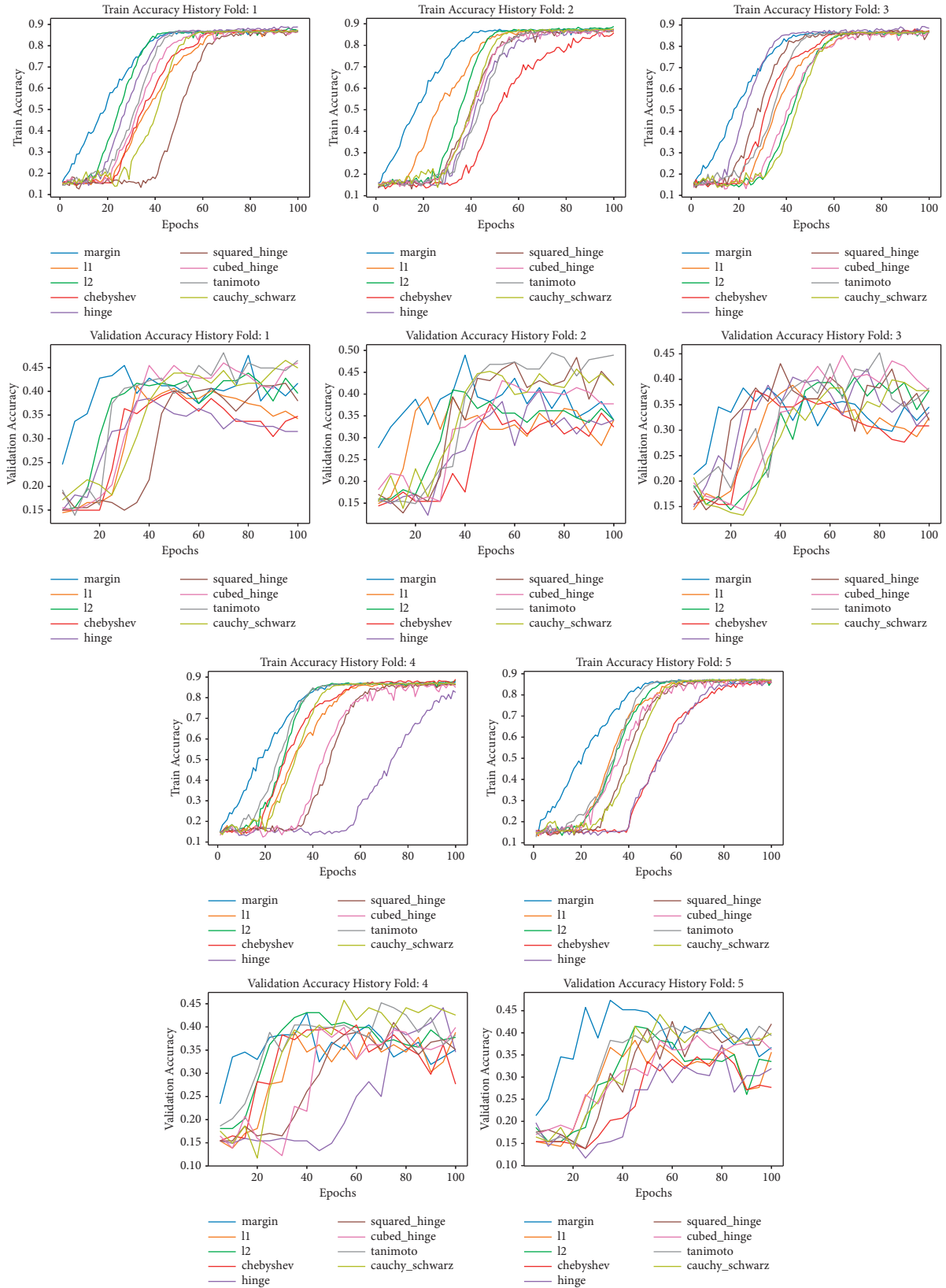


FIGURE 4: CAFE train and validation accuracy history for capsule model and different loss functions.

TABLE 4: CAFE class F1 scores and overall accuracies.

Loss type	Anger (F1)	Disgust (F1)	Happiness (F1)	Neutral (F1)	Fear (F1)	Surprise (F1)	Sadness (F1)	Overall (F1)	Overall (Acc.)
Margin loss	45.07	42.57	42.61	14.44	47.72	51.38	52.20	42.28	45.84
L1 loss	37.88	41.68	33.92	16.83	36.82	50.57	46.28	37.71	40.78
L2 loss	39.80	39.82	39.96	29.49	40.64	49.84	43.27	40.40	41.67
Chebyshev loss	41.64	38.83	24.90	5.22	35.01	44.70	48.73	34.15	38.57
Hinge loss	41.83	37.42	30.89	22.62	41.92	49.03	43.38	38.16	40.92
Square hinge loss	46.85	45.35	37.52	24.89	44.75	49.22	47.08	42.24	44.66
Cubed hinge loss	45.54	42.32	34.47	31.82	40.81	45.09	47.09	41.02	42.20
Tanimoto loss	49.57	43.06	38.06	31.07	46.89	54.83	48.28	44.54	46.36
Cauchy–Schwarz loss	49.58	43.53	35.72	42.03	48.79	52.46	51.82	46.28	47.01

The configuration used for the capsule network used in this paper is exactly described by Sabour et al. [8] and is shown in Figure 1. An initial convolution layer with 256 filters of size 9 and stride 1 extracts features from the image inputs. After the initial CNN layer, a PrimaryCaps layer with 256 channels from 32 8-dimensional capsules of size 9 and stride 2 follows. The last layer will differ in the number of capsules based on the number of unique classes in the data set. Each capsule in this last layer has 16 dimensions. The Adam optimizer is used with a learning rate of 0.001 and betas equal to 0.9 and 0.999, respectively. A decoder is also used to add in a reconstruction loss as a regularization term. The baseline model uses the margin loss described in the original literature.

In contrast, the other comparative models will use the other loss functions, with the rest of the architecture staying the same. Since the capsule architecture works best with image inputs, the input sequence for the network are time-frequency spectrograms extracted from the speech samples. Each spectrogram is a 64×64 image, unlike the 28×28 images from the MNIST data set. The data sets were divided into a 2 : 1 : 1 split with the larger split for the training set and the other two splits for the validation and test sets. The models were cross-validated on five-folds for 100 epochs with a validation step every five epochs. After the training stage in each fold, the highest validation accuracy model would be used for the test set. For the training sets, some data augmentation, such as noise injection and voice tract length perturbation (VTLP).

4. Results and Discussion

In each data set, the training and validation accuracies are logged and graphed in the course of 100 epochs. In addition, the F1 scores for each emotion class and overall accuracies are shown in the tables below.

4.1. The Analysis for Different Loss Functions. For the first data set RAVDESS, a few remarks can be observed from the data in Figure 2 and Table 2 regarding the loss functions. The original margin loss remains the fastest in learning among the loss functions reaching more than 80% train accuracy at around 40 epochs. L2 loss also seems to be a considerable choice for a faster learning speed but with a less significant

difference from the following loss function. The Cauchy–Schwarz divergence loss function learns slowly but lessens overfitting as observed on the validation accuracy histories. The Cauchy–Schwarz divergence and Tanimoto losses are the top two loss functions on F1 and accuracy. Both loss functions greatly improved on the baseline for almost all the individual classes, including the minority neutral emotion. The reason for this might be that these two loss functions consider the similarity of the compared vectors from the perspective of set theory. Unsurprisingly, these same two loss functions also perform pretty well in Janocha and Czarnecki’s study [15]. Also mentioned by Janocha and Czarnecki [15] is that Cauchy–Schwarz divergence performs as well as cross-entropy loss or log loss in terms of learning speed and final performance.

Two loss functions performed the worst in EMODB. As shown in Figure 3 and Table 3, they are the L1 loss and Chebyshev loss. For samples that have been classified as correct, the individual elements of the target and predicted vectors might still be considerably different, which will still lead to a massive penalty during optimization. The penalty is amplified even further when using Chebyshev loss as even a correct classification may still lead to a higher loss. Out of the four data sets, only EMODB produced results where the baseline, margin loss, remained the best. One major cause for this result is the lack of sufficient samples in EMODB. Even with data augmentation, the newly generated samples may still resemble the original audio sample.

As shown in Figure 4 and Table 4, margin loss remains the fastest among the loss functions on the CAFE data set. Owing to the values of m^+ and m^- being specifically chosen for the capsule network after rigorous experimentation by the original authors, it is not a surprise that the loss function would be highly optimized. The validation accuracy histories of the different loss functions present constant shifting, which means that model can no longer improve on the validation set. The constant shift can easily be an easy sign of overfitting and a signal for early stopping. In terms of accuracy, the two best loss functions are still Tanimoto and Cauchy–Schwarz, albeit with a less significant lead on the baseline. Among the maximum-margin based losses, only squared hinge was able to perform as well as the baseline. It also did the best on the minority class, which is disgust. Perhaps the order of this hinge loss function is just in the right spot to not amplify significant errors and minimize minor errors.

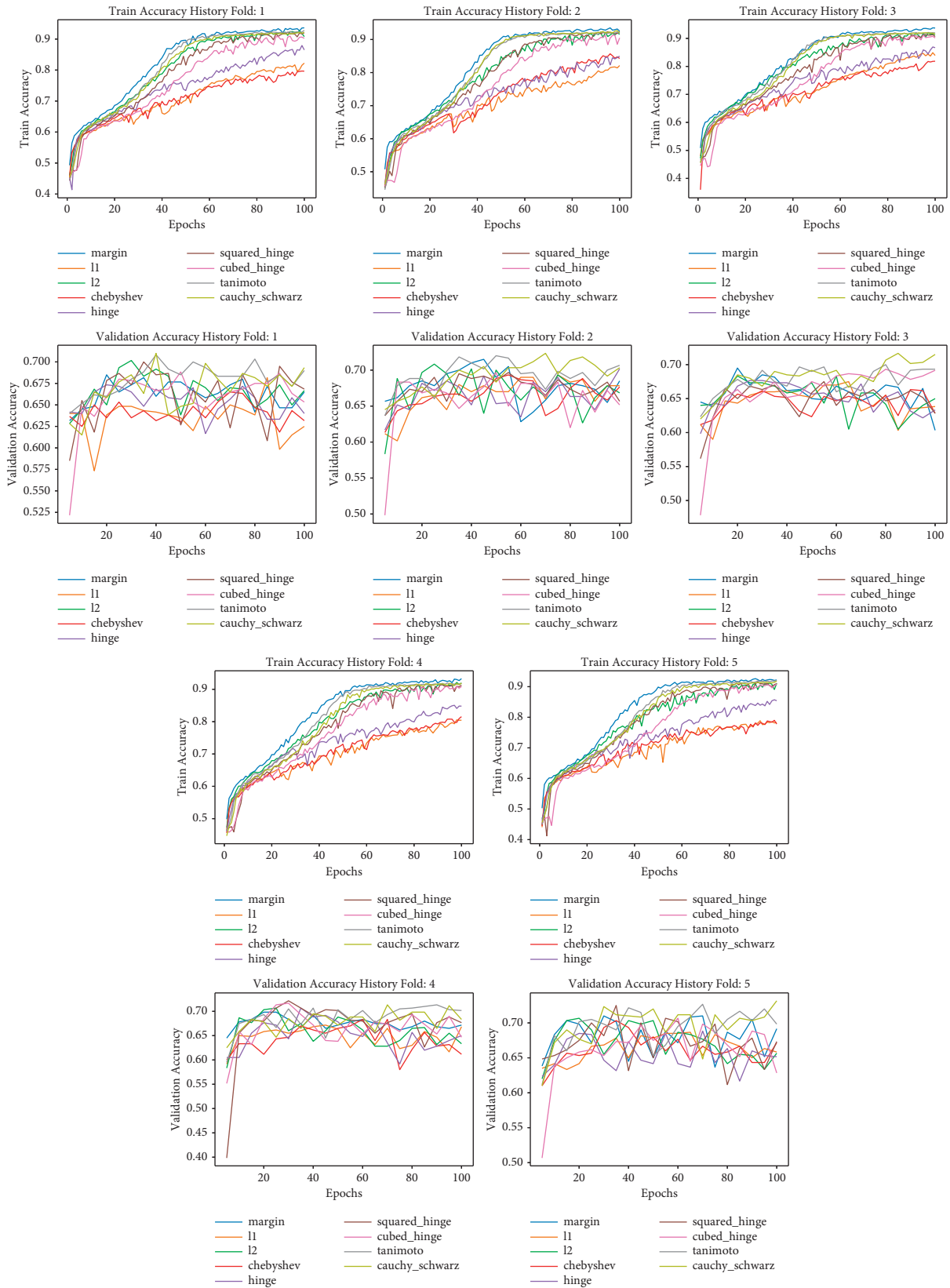


FIGURE 5: SHEMO train and validation accuracy history for capsule model and different loss functions.

TABLE 5: SHEMO class F1 scores and overall accuracies.

Loss type	Anger (F1)	Fear (F1)	Happiness (F1)	Neutral (F1)	Sadness (F1)	Surprise (F1)	Overall (F1)	Overall (Acc.)
Margin loss	79.60	0.00	14.65	76.69	50.99	52.03	45.66	69.70
L1 loss	78.07	0.00	4.34	74.67	53.02	32.11	40.37	68.10
L2 loss	80.37	0.00	17.69	77.45	55.39	50.89	46.97	70.43
Chebyshev loss	77.43	0.00	0.95	74.71	50.28	45.09	41.41	67.63
Hinge loss	78.58	0.00	3.33	76.25	52.61	48.13	43.15	69.27
Square hinge loss	80.58	4.00	26.82	76.96	49.04	51.93	48.22	70.17
Cubed hinge loss	78.90	0.00	19.21	78.06	52.18	47.23	45.93	69.87
Tanimoto loss	81.15	0.00	26.73	78.47	56.52	51.71	49.10	71.43
Cauchy-Schwarz loss	80.55	4.44	25.42	77.80	54.52	57.26	50.00	71.06

TABLE 6: Comparison with previous works (unweighted accuracies).

Model	Data set			
	RAVDESS (%)	EMODB (%)	CAFE (%)	SHEMO (%)
Capsule	69.03	73.46	47.01	71.43
CNN-BiGRU [7]	70.07	66.92	44.13	67.47
Head fusion [28]	57.85	68.04	41.45	70.60
LSTM [29]	68.19	71.59	48.18	69.77

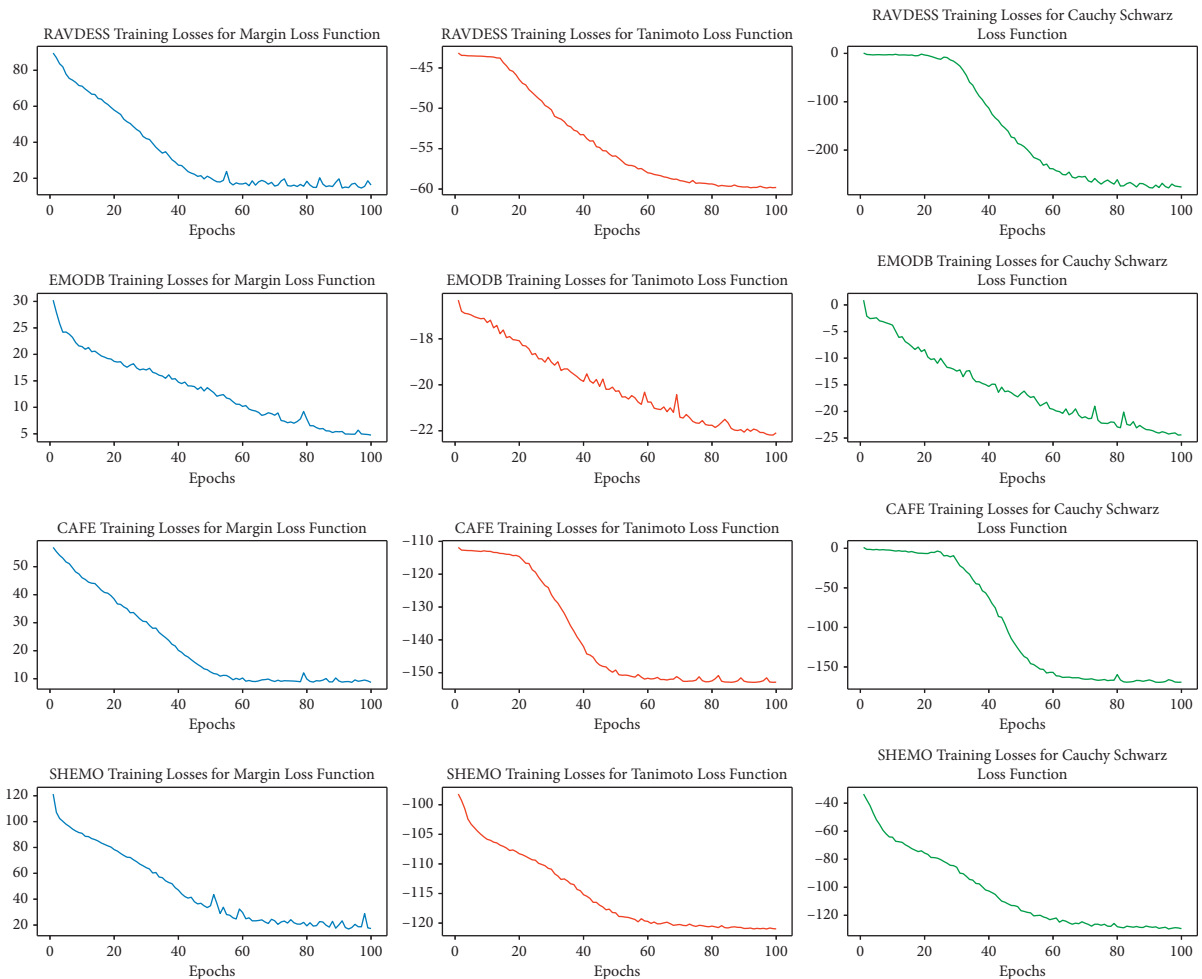


FIGURE 6: Training losses in a single fold for margin, Tanimoto, and Cauchy-Schwarz loss functions.

Figure 5 and Table 5 show the results for the SHERO data set. The first thing that is relatively clear from Table 5 is low scores under the fear class with only 38 samples. Despite that the model was able to achieve an accuracy of 71% with the Tanimoto loss. Both Tanimoto and Cauchy-Schwarz divergence losses once again performed the best. Significant improvements were observed in the minority classes, such as fear, happiness, sadness, and surprise. If measures were to be taken to address the imbalance problem, the accuracy might increase, but the effect of these two losses might be less significant instead.

Finally, three more baseline models are implemented from other works for comparison. The first model is a combination of a CNN and a bidirectional gated recurrent unit network (BiGRU) model with focal loss function proposed by Zhu et al. [7]. In this model, the spectrogram features are passed through the CNN, after their temporal properties are analysed by the BiGRU. Next, the second model is a CNN model with a custom attention mechanism called head fusion [28], which is based on multihead attention. Finally, the third model is an LSTM model with a regular attention mechanism as described by Xie et al. [29]. All the baseline models use the same set of features as the capsule network. As shown in Table 6, the best capsule network accuracy is taken and compared with the previous works. Across the data sets, the capsule network performs as well as an LSTM especially on the EMODB data set. The ability of the capsule to encode spatial information would most likely complement well with an LSTM's affinity for encoding temporal information. The combination of both can be a good new research direction to consider. Another mechanism to consider is an attention mechanism, but its addition can be highly redundant to the dynamic routing.

4.2. Convergence Analysis for Tanimoto, Cauchy-Schwarz, and Margin Loss. To provide a better understanding for the reason of the Tanimoto and Cauchy-Schwarz loss functions' better performance, the training losses (in a single fold) for each type of loss are plotted as shown in Figure 6. It is clear in the RAVDESS data set that Tanimoto and Cauchy-Schwarz perform better because they converge a bit later than margin loss. On other data sets, the performances of Tanimoto and Cauchy-Schwarz in comparison with Margin loss are relatively similar; hence, they have similar curves and converge at roughly similar times. One thing to also note is that Tanimoto and Cauchy-Schwarz on both RAVDESS and CAFE data sets do not immediately have lowering losses within the first 20 epochs. This may mean that these loss functions are taking their time in learning in the initial portion of training.

5. Conclusion

This paper analyses the use of a capsule network and several different loss functions on SER data sets. Results showed that Tanimoto and Cauchy-Schwarz losses can highly improve capsule network performance by improving on the minority classes. Comparisons of the capsule network with previous

deep learning models in the field also show that the capsule network performs marginally better. Future research directions will experiment on the use of capsule networks combined with LSTMs to use both their capabilities in learning spatial and temporal information, respectively.

Data Availability

The data are available at <https://zenodo.org/record/1188976#> (RAVDESS), <https://zenodo.org/record/1478765#> (CAFE), and <https://github.com/mansourehk/ShEMO> (ShEMO).

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (grant no. 61772023) and the National Key Research and Development Program of China (grant no. 2019QY1803).

References

- [1] N. Fragopanagos and J. G. Taylor, "Emotion recognition in human-computer interaction," *Neural Networks*, vol. 18, no. 4, pp. 389–405, 2005.
- [2] K. Vicsi and D. Sztahó, "Emotional state recognition in customer service dialogues through telephone line," in *Proceedings of the 2011 2nd International Conference on Cognitive Infocommunications (CogInfoCom)*, Budapest, Hungary, 2011.
- [3] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.
- [4] D. Issa, M. Fatih Demirci, and A. Yazici, "Speech emotion recognition with deep convolutional neural networks," *Biomedical Signal Processing and Control*, vol. 59, Article ID 101894, 2020.
- [5] D. Li, Y. Zhou, Z. Wang, and D. Gao, "Exploiting the potentialities of features for speech emotion recognition," *Information Sciences*, vol. 548, pp. 328–343, 2021.
- [6] B. T. Atmaja and M. Akagi, "Speech emotion recognition based on speech segment using LSTM with attention model," in *Proceedings of the 2019 IEEE International Conference on Signals and Systems (ICSigSys)*.
- [7] Z. Zhu, W. Dai, Y. Hu, and J. Li, "Speech emotion recognition model based on Bi-GRU and focal loss," *Pattern Recognition Letters*, vol. 140, pp. 358–365, 2020.
- [8] S. Sabour, N. Frosst, and G. Hinton, "Dynamic routing between capsules," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, Long Beach, CA, USA, December 2017.
- [9] H. Yao, Y. Tan, C. Xu, J. Yu, and X. Bai, "Deep capsule network for recognition and separation of fully overlapping handwritten digits," *Computers & Electrical Engineering*, vol. 91, Article ID 107028, 2021.
- [10] J. Poncelet, V. Renkens, and H. Van hamme, "Low resource end-to-end spoken language understanding with capsule networks," *Computer Speech & Language*, vol. 66, Article ID 101142, 2020.

- [11] P. Afshar et al., “COVID-CAPS: a capsule network-based framework for identification of COVID-19 cases from x-ray images,” 2020, <https://arxiv.org/abs/2004.02696>.
- [12] K. Lei, Q. Fu, and Y. Liang, “Multi-task learning with capsule networks,” in *Proceedings of the 2019 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8.
- [13] H. Yao, P. Zhang, C. Jiang, and Z. Han, “Capsule network assisted IoT traffic classification mechanism for smart cities,” *IEEE Internet of Things Journal*, vol. 6, pp. 7515–7525, 2019.
- [14] B. T. Atmaja and M. Akagi, “Evaluation of error- and correlation-based loss functions for multitask learning dimensional speech emotion recognition,” *Journal of Physics: Conference Series*, vol. 1896, no. 1, Article ID 012004, 2021.
- [15] K. Janocha and W. Czarnecki, “On loss functions for deep neural networks in classification,” *Schedae Informaticae*, vol. 25, 2017.
- [16] S. Langari, H. Marvi, and M. Zahedi, “Efficient speech emotion recognition using modified feature extraction,” *Informatics in Medicine Unlocked*, vol. 20, Article ID 100424, 2020.
- [17] L. Kerkeni, Serrestou, Raouf, M. Mbarki, A. A. Mahjoub, and C. Cleder, “Automatic speech emotion recognition using an optimal combination of features based on EMD-TKEO,” *Speech Communication*, vol. 114, pp. 22–35, 2019.
- [18] T. Özseven, “A novel feature selection method for speech emotion recognition,” *Applied Acoustics*, vol. 146, pp. 320–326, 2019.
- [19] Z. Yao, Y. Liu, and J. Pan, “Speech emotion recognition using fusion of three multi-task learning-based classifiers: HSF-DNN, MS-CNN and LLD-RNN,” *Speech Communication*, vol. 120, pp. 11–19, 2020.
- [20] J. Zhao, M. Mao, and L. Chen, “Speech emotion recognition using deep 1D & 2D CNN LSTM networks,” *Biomedical Signal Processing and Control*, vol. 47, pp. 312–323, 2019.
- [21] K. Lee, H. Joe, H. Lim et al., “Sequential routing framework: fully capsule network-based speech recognition,” *Computer Speech & Language*, vol. 70, Article ID 101228, 2021.
- [22] X. Wu, S. Liu, Y. Cao et al., “Speech emotion recognition using capsule networks,” in *Proceedings of the ICASSP 2019*—, p. 5.
- [23] R. Jain, “Improving performance and inference on audio classification tasks using capsule networks,” 2019, <https://arxiv.org/abs/1902.05069>.
- [24] S. R. Livingstone and F. A. Russo, “The Ryerson audio-visual database of emotional speech and Song (RAVDESS): a dynamic, multimodal set of facial and vocal expressions in North American English,” *PloS One*, vol. 13, no. 5, Article ID e0196391, 2018.
- [25] F. Burkhardt, W. Sendlmeier, and B. Weiss, “A database of German emotional speech,” in *Proceedings of the INTERSPEECH 2005 - Eurospeech, 9th European Conference on Speech Communication and Technology*, p. 4, liston, portugal.
- [26] P. Gournay, O. Lahaie, and R. Lefebvre, “A canadian French emotional speech dataset,” in *the 9th ACM Multimedia Systems Conference*, pp. 399–402, Amsterdam, Netherland, June 2018.
- [27] O. Mohamad Nezami, P. Lou, and M. Karami, “ShEMO—a large-scale validated database for Persian speech emotion detection,” *Language Resources and Evaluation*, vol. 53, 2019.
- [28] M. Xu, “Head fusion: improving the accuracy and robustness of speech emotion recognition on the IEMOCAP and RAVDESS dataset,” *IEEE Access*, vol. 9, pp. 74539–74549, 2021.
- [29] Y. Xie, R. Liang, Z. Liang, C. Huang, C. Zou, and B. Schuller, “Speech emotion classification using attention-based LSTM,” *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 27, no. 11, pp. 1675–1685, 2019.

Research Article

How Many Bedrooms Do You Need? A Real-Estate Recommender System from Architectural Floor Plan Images

Y.S. Gan ¹, Shih-Yuan Wang,² Chieh-En Huang,³ Yi-Chen Hsieh,³ Hsiang-Yu Wang,³ Wen-Hung Lin,³ Shing-Nam Chong,³ and Sze-Teng Liong ³

¹*School of Architecture, Feng Chia University, Taichung, Taiwan*

²*Graduate Institute of Architecture, National Chiao Tung University, Hsinchu, Taiwan*

³*Department of Electronic Engineering, Feng Chia University, Taichung, Taiwan*

Correspondence should be addressed to Sze-Teng Liong; stliong@fcu.edu.tw

Received 7 March 2021; Revised 16 June 2021; Accepted 28 July 2021; Published 6 August 2021

Academic Editor: Jianping Gou

Copyright © 2021 Y.S. Gan et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

This paper introduces an automated image processing method to analyze an architectural floor plan database. The floor plan information, such as the measurement of the rooms, dimension lines, and even the location of each room, can be automatically produced. This assists the real-estate agents to maximise the chances of the closure of deals by providing explicit insights to the prospective purchasers. With a clear idea about the layout of the place, customers can quickly make an analytical decision. Besides, it reduces the specialized training cost and increases the efficiency in business actions by understanding the property types with the greatest demand. Succinctly, this paper utilizes both the traditional image processing and convolutional neural networks (CNNs) to detect the bedrooms by undergoing the segmentation and classification processes. A thorough experiment, analysis, and evaluation had been performed to verify the effectiveness of the proposed framework. As a result, a three-class bedroom classification accuracy of ~ 90% was achieved when validating on more than 500 image samples that consist of the different room numbers. In addition, qualitative findings were presented to manifest visually the feasibility of the algorithm developed.

1. Introduction

When designing a building, the most indispensable tool for the architect is the floor plan, which also served as an important element to provide building guidelines and instructions for construction. In brief, a floor plan demonstrates the relationships between rooms, spaces, and other physical characteristics in a visual form. The floor plan usually specifies the basic layout dimensions (i.e., room size, height, and length) with an annotated scale factor. The architects often utilize several symbols or icons to enhance the interpretation of the design of a floor plan, for instance, simple outlines to indicate the features of walls, windows, and door openings. Besides, the floor plan suggests the arrangement of space by including the details of fixtures and furniture like the stove, bathtub, sofa, and toilet bowl. Nonetheless, the floor plan design is not limited to housing; it is applicable to any building type such as the office and Church.

Despite the fact that there are standard architectural symbols used to represent common building components and features, architectural drafting can basically be categorized into three types: process drawings (sketches/schematics), construction documents (drafted drawings), and presentation drawings (illustrated sketches). However, some architects artistically add visual interest to the floor plan to convey the intended idea and improve the understanding via graphic language. Although there are design guidelines for the sketching and drafting of a floor plan, it is subjective to the individual as the design shows some room for creativity and flexibility to attract the attention of viewers. Since there are plenty of design alternatives for the appearance of a floor plan, it requires substantial effort in rendering, extracting, learning, and recognizing the semantic information from the human perspective. Therefore, it increases the difficulty when processing and interpreting the floor plans using image analysis techniques.

To the best of our knowledge, this is the first attempt to comprehensively address and analyze the details of the rooms from the floor plans. In this paper, a benchmark framework is provided to automatically determine the location and the number of bedrooms from a floor plan. Following the significant success with deep learning frameworks, there has been a surge of interest resolving in computer vision-related problems. For instance, inspirational applications developed recently, such as biomedical field [1], unmanned aerial vehicle navigation [2], facial expression recognition [3], and soil structure assessment [4]. Thus, motivated by these works, several deep learning approaches are applied in the bedroom detection task herein to facilitate the automatic computational image processing analysis.

In summary, the main contributions of this paper are highlighted briefly as follows:

- (1) Application of a series of preprocessing techniques to improve the image quality, such as noise removal, wall thickness adjustment, and image scaling.
- (2) Proposal of a visual understanding mechanism to distinguish the bedroom from the floor plan by implementing the segmentation and classification processes.
- (3) Comprehensive experimentation on the dataset to verify the effectiveness of the algorithms evaluated. Both the qualitative and quantitative results are presented.

The remainder of the paper is organized as follows: a review of related literature is presented in Section 2. Subsequently, Section 3 describes the complete framework in detail, including the preprocessing method, the configuration of experiment settings, and performance metrics for result validation. Section 5 reports and analyzes the experimental results. Finally, conclusions are drawn in Section 6.

2. Related Work

A residential floor plan is a key element to provide the prospective buyer the essence of the property regarding the internal amenity, the outlook, and the interaction of the spaces that are viewed from above. The floor plan serves as the most essential guide for the home buyer to consider purchasing the property [5]. An eye-catching and expressive floor plan contains a colorful design, accurate scale, basic furniture icons, and a clear flow of spaces. A residential project has a briefer form and contains lesser information compared to a commercial project as it is a simple diagram that offers a conceptual starting point. Note that there is no standard real-estate floor plan and thus the oversimplified or overprofessional design may cause confusion to the buyer. Therefore, there is a lack of an automatic system to relate the architectural design to computer vision technology. Specifically, this automatic task is useful in assisting the buyers to quickly determine the number of bedrooms in each floor plan, classify the space according to the floor plan, analyze

the amount of space in a floor plan, and determine the exact locations for each room.

Albeit the emergence of artificial intelligence, the existing literature studies in analyzing the architectural designs with this technology are manageably finite. For example, Bayer et al. [6] suggested a semiautomatic method to automatically generate the floor plans of specific buildings. Concretely, the long short-term memory (LSTM) [7] was utilized as predictive modeling to achieve this task. However, instead of passing the floor plan images to the suggested model structure, the input information requires manual human effort to extract certain information from each sample image. Besides, the trained model insufficiently describes the detailed contextual characteristics of the floor plan, such as the actual position of the basic building blocks (i.e., walls, doors, and windows). A similar work that performs the floor plan generation is conducted by [8]. Yet, a deeper understanding of the current development state of image processing approaches is integrated to resolve for the best placement solution. The system is built on the Grasshopper environment to make the user interface design legible and easy to use.

On the other hand, Liu et al. [9] proposed a novel convolutional neural network (CNN) architecture, namely FloorNet, to reconstruct the 3D floor plans by physically scanning the indoor spaces over a visual display. Concretely, a triple-branch hybrid design is implemented to simultaneously process the 3D coordinate points, 2D floor plan, and images captured, to form the final floor plan. Thus, a dataset that comprises ~ 155 residential house layouts has been created. This work provides a reverse engineering solution on the floor plan image, whereby they did not process on the existing images; instead, they introduce a series of image analyses to generate a new floor plan image. The main intuition of this work is to cope with the absence of floor plan design problem, especially in the region like North America. Recently, an improved method that can generate the floor plan via 3D scanning with higher accuracy and higher speed is introduced by [10]. Specifically, the proposed pipeline, namely, Scan2Plan demonstrated outstanding results when evaluated on the publicly available Structured3D [11] and BKE [12] datasets.

De las Heras et al. [13] designed a structural element recognition system to detect the wall and room from four datasets with different ground-truth characteristics. The proposed approach first extract the main entities using statistical segmentation, such as the walls, door, and windows. Then, the structural compositions of the building are identified and distinguished using an image recognition technique. The authors claimed that the proposed algorithm is adaptable to any graphical representation, as it can extract the structural elements without prior knowledge of the modeling conventions of the plan. Nevertheless, further analysis and classification regarding the type or function of the room are not presented in the work. On the other hand, Khade et al. [14] focused on extracting the outer shape of the floor plan. A series of algorithms is suggested, in which geometric features such as area, corners, quadrants, distance, slope, and angle are involved. To evaluate the robustness of

the proposed framework, the experiments are tested on both the original and rotated images of a synthetic dataset.

Besides, [15] introduced a method to detect the elements of the walls and identify the key objects, as well as determining the characters from the floor plan images. The methodologies to accomplish this task are to adopt a fully convolutional network (FCN) model and an optical character recognition (OCR) technique. In brief, OCR is to retrieve meaningful room labeling. Experiments were carried out on two datasets, namely, a publicly available dataset (CVC-FP) and self-collected from the real-estate website. The experiments were performed on the datasets were wall segmentation, object detection (door, sliding door, kitchen oven, bathtub, sink, and toilet), and text recognition. Although promising wall segmentation was reported in the paper, the proposed approach did not compare to the state of the art. Besides, the number of testing samples to evaluate the object detection and text recognition performance was relatively few (<50 images). Thus, this work does not provide conclusive empirical evidence to verify the effectiveness of the proposed method.

On another note, Ahmed et al. [16] recognized the room by differentiating the walls with different thicknesses (i.e., thick, medium, and thin). They also pointed out that the wall thickness (i.e., thin wall) greatly affected the wall detection and hence influenced the recognition performance. The following year, the same research group [17] presented an automatic system to analyze and label the architectural floor plans. Particularly, the SURF [18] feature descriptor is utilized to spot the building element in enhancing semantic information extraction. This paper extends the previous paper by dividing the rooms into several subpartitions for the cases of a shared room. However, there is some false detection especially when the text touched the graphics component. Later, Ahmed et al. [19] separated the wall and text in the image using simple morphological erosion and dilation techniques. As a result, a promising result of the recall of 99% is exhibited when evaluating ~ 90 images. The prevalence performance mostly contributed to the success in removing the thin lines especially the text touching the lines.

Recently, Goyal et al. [20] proposed SUGAMAN (Supervised and Unified framework using Grammar and Annotation Model for Access and Navigation) to briefly describe the indoor environment in natural language from the building floor plan images. They represented the room features by adopting a local orientation and frequency descriptor (LOFD). Then, a single-layer neural network with 10 neurons is employed to learn the room annotations for room classification. To examine the effectiveness of the proposed algorithm, experiments are conducted on a dataset with more than 1000 image samples. Results demonstrated that the proposed method outperformed the state of the art by attaining higher classification accuracy when identifying the decor items (i.e., bed, chair, table, sofa, etc.). However, there are some scenarios where the proposed method incorrectly classifies the type of the room (i.e., kitchen vs. room). Besides, the result of floor plan creation based on LOFD is discouraging.

3. Proposed Method

The proposed algorithm aims to determine the existence of the bedroom and its respective location, as well as computing the number of bedrooms from an architectural floor plan image. Figure 1 lists the proposed framework of the proposed method. In brief, it incorporates four primary stages: wall extraction, wall thickening and door gap closing, room partitioning and decor item retrieval, and bedroom classification. The details of each step are discussed in this section by providing greater details in terms of their respective mathematical derivations and pseudocodes. For better visualization, Figure 2 depicts the conceptual diagram for each stage. Note that, the dataset involved in the experiment is Robin because the decor items shown on the images are clear and do not contain any text information.

3.1. Wall Extraction. The first step is to acquire the wall lines whilst removing the decor items. Prior to that, all the images are converted from RGB colorspace to grayscale. The original RGB image is shown in Figure 3(a) and the grayscale image is illustrated in Figure 3(b). Then, Otsu's algorithm is employed to perform the thresholding technique to enhance image contrast. Otsu's algorithm is one of the simplest methods to categorize the pixel intensity into two classes and the output is generated by minimizing the intraclass variance of each image. It is noticed that the decor items are appearing in thinner lines compared to the walls. Therefore, a simple morphological image processing method, namely, closing, is adopted to remove the objects with thin lines.

The closing morphology operation of image A by the structuring element B is the composite of erosion and dilation operations, defining as follows:

$$A \cdot B = (A \oplus B) \ominus B, \quad (1)$$

where $X \oplus Y$ denotes the dilation operation of image X by the structuring element Y . On the other hand, $X \ominus Y$ denotes the erosion operation of image X by the structuring element Y . In brief, the dilation and erosion operations can be defined as follows, respectively:

$$\begin{aligned} X \oplus Y &\equiv \{(x + y) \mid \forall x \in X, y \in Y\}, \\ X \ominus Y &\equiv \{x \in \mathbb{Z}^2 \mid (x + y \in X, \forall y \in Y)\}. \end{aligned} \quad (2)$$

By doing so, the decor items can be eliminated, in the meantime, preserving the wall lines, as illustrated in Figure 2(b). Besides, Figure 3(c) shows the output image after extracting the wall lines on another sample, whereby lesser furniture and elements exist in this 3-bedroom image. Since the decor items are represented by a set of simple synthetic architectural symbols made up of straight lines and simple shapes, this wall extraction process occupies relatively lesser computational cost and algorithm complexity.

3.2. Wall Thickening and Door Gap Closing. As there are noticeable door gaps after performing the wall extraction in the previous step (i.e., Figure 2(b)), the gaps are filled by

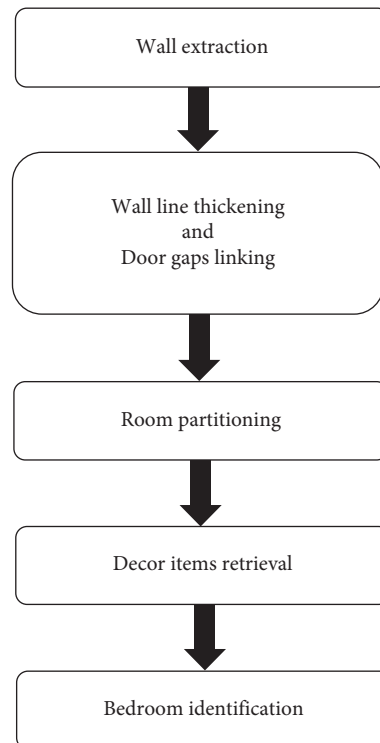


FIGURE 1: The proposed pipeline incorporates five stages.

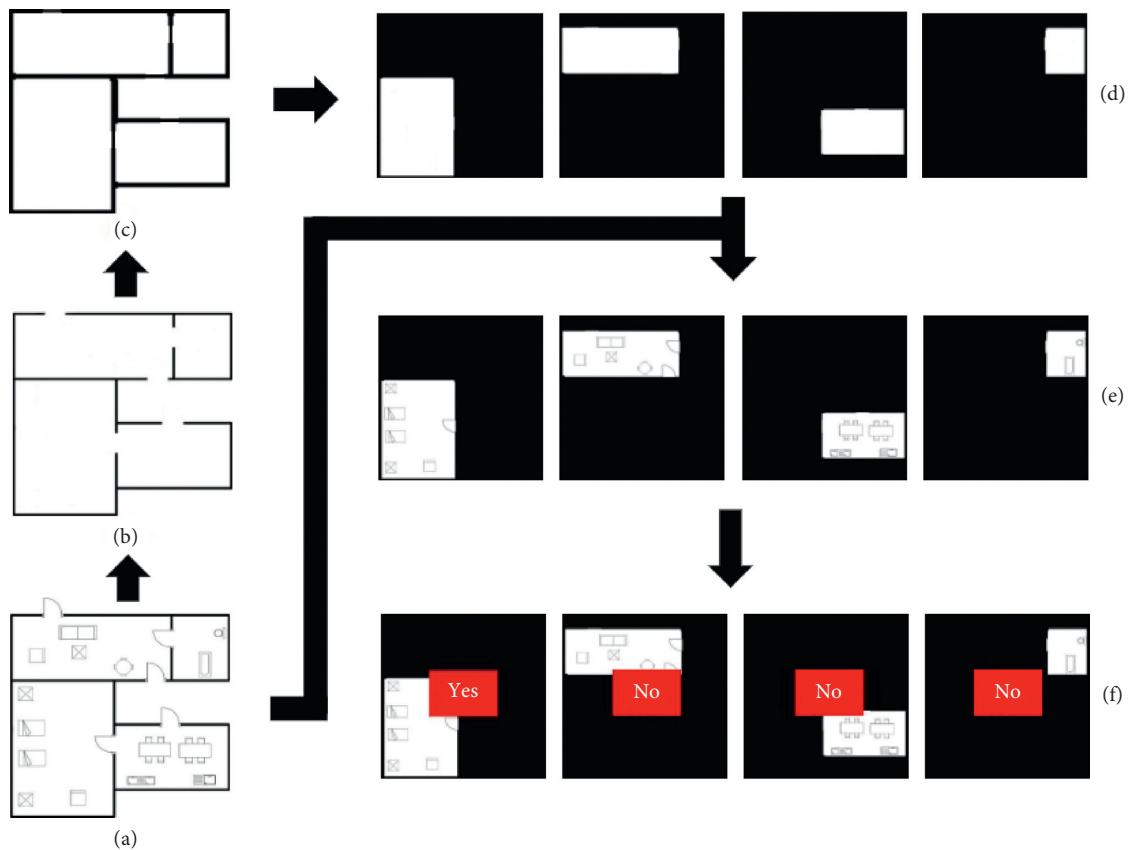


FIGURE 2: Overall process flow illustration: (a) original dataset; (b) wall extraction; (c) wall line thickening and door gaps linking; (d) room partitioning; (e) decor items retrieval, and (f) bedroom identification.

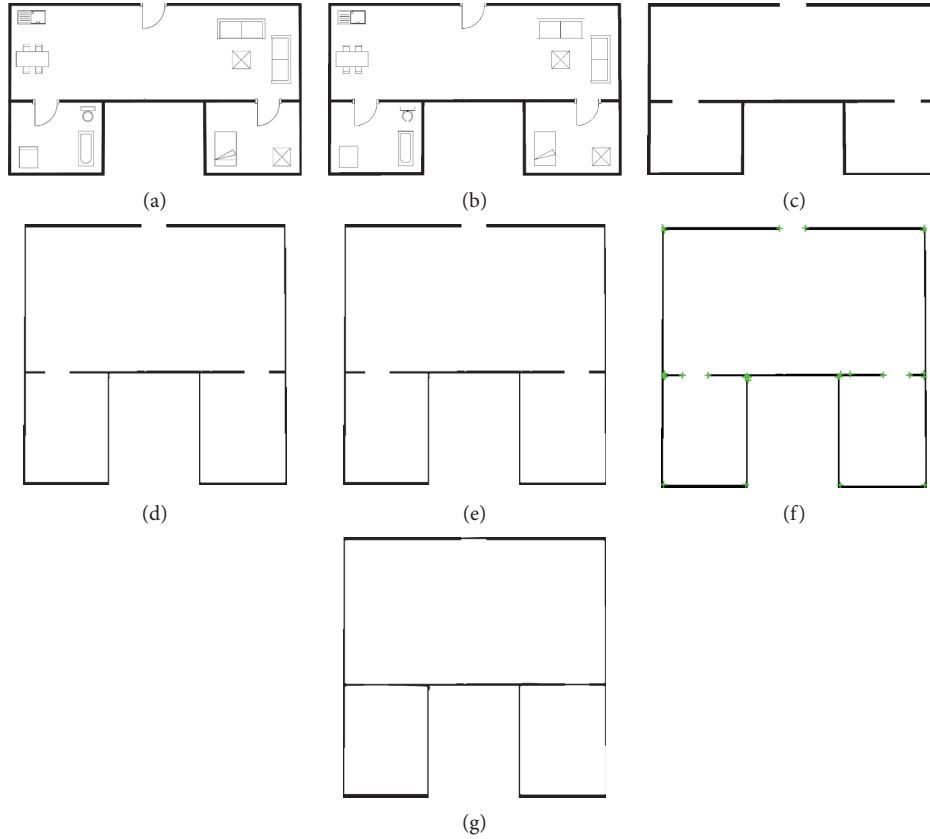


FIGURE 3: Output generated after performing each image transformation process: (a) original image; (b) pixel binarization; (c) wall extraction; (d) image dilation; (e) image resize to 400×400 ; (f) edge detection using FAST algorithm, and (g) door gaps linking.

thickening the walls to facilitate the gaps connection. A dilation morphological process is performed to improve the visual appearance of the wall lines, as shown in Figure 3(d). The dilation operation is expressed as

$$A \oplus B = \bigcup_{b \in B} A_b. \quad (3)$$

Since the resolution for the images in the Robin dataset is varied, the size is standardized such that the height and width are having the same length, namely, 400×400 , as shown in Figure 3(e). Next, the pairs of the door gaps coordinates for the linkage are identified. It can be achieved by applying Hough transform [21] and the FAST algorithm for edge detection. Subsequently, the width and the exact position of the door gaps are obtained, as shown in Figure 3(f).

The algorithm details to acquire the wall lines information are shown in Algorithm 1. In brief, there are three inputs required to link the door gaps, namely, the position, width, and the total number of the door gaps. Lines 1 and 2 in Algorithm 1 scan all the pixels in the image. Line 3 compares two coordinates to decide if they are the pair of the door gap, depending on the predefined width parameter. Lastly, line 4 links the pair of door gap coordinates, as shown in Figure 3(g).

Figure 2(c) shows the output of the door gaps linking after increasing the thickness of the wall.

3.3. Room Partition and Decor Items Retrieval. This step is to split the multiple rooms from each floor plan into individual rooms. Since the floor plan images in Robin are generated by the computer, it is found that there is some absurdly small size of rooms in a few floor plan images. Therefore, if the detected room region occupies less than 300 pixels, it will be eliminated from the next processing step. Figure 2(d) illustrates the segmented rooms from a floor plan.

After identifying and filtering the rooms, the rooms are overlapped with the original floor plan image, such that the decor items appear in every single room. Figure 2(e) illustrates the insertion of the original decor items on the respective partitioned rooms. The pseudocode to realize the room partition and decor items retrieval steps is shown in Algorithm 2. In brief, lines 1 to 27 detect the single rooms by identifying the pixel intensity values. A room should be encircled by a continuous pixel value of 0 in a rectangle shape. Concretely, the program will scan the entire image from left to right and top to bottom, to search for the first appeared pixel intensity value of 255. Then, that particular pixel is arbitrarily set to T , where T is initially defined as 1. This specific pixel is treated as a reference point such that the pixel values of all the four directions (i.e., top, bottom, left, and right) are set to 1. This pixel value updating step will be terminated until the pixel value 0 is met. Consequently, another scanning to search for the next pixel with an intensity of 255 will be performed. If found, that particular

```

Input:  $D(i, :), D(j, :)$  ← coordinate points of the door gaps
 $W$  ← width of the door gaps
 $N$  ← total number of door gaps
Output: the door gaps are closed and independent rooms can be identified
(1) for  $i = 1; i \leq N; i++$  do
(2)   for  $j = 1; j \leq N; j++$  do
(3)     if  $|D(i, :) - D(j, :)| \leq W$  then
(4)       Link  $D(i, :)$  and  $D(j, :)$ 
(5)     else
(6)       break.

```

ALGORITHM 1: Door gap closing.

```

Input:
 $A(i, j)$  ← coordinate point of an independent room image
 $m$  ←  $x$ -axis image size of an independent room
 $n$  ←  $y$ -axis image size of an independent room
Output:
 $T$  ← number of independent rooms
(1)  $v = 1;$ 
(2)  $T = 1;$ 
(3) While  $v \neq 0 \triangleright v = 1$  indicates that a new area has been found
(4) do
(5)    $v = 0;$ 
(6)    $b = 0;$ 
(7)   for  $i = 1; i \leq n; i++$  do
(8)     for  $j = 1; j \leq m; j++$  do
(9)       if  $A(i, j) == 255$   $\triangleright$  pixel intensity of 255 indicates white
(10)      then
(11)         $A(i, j) = T;$ 
(12)         $v = 1;$ 
(13)        break
(14)    $Q = 1;$ 
(15)   While  $Q \neq 0 \triangleright Q = 1$  indicates there may be missing points
(16)   do
(17)      $Q = 0;$ 
(18)     for  $i = 1; i \leq n; i++$  do
(19)       for  $j = 1; j \leq m; j++$  do
(20)         If  $A(i, j) == T$  then
(21)           Convert the pixel values to 255 in the top, bottom, left, and right directions of points  $A(i, j)$  into  $T$ 
(22)           This process is terminated when it encounters 0.
(23)         if a pixel is converted then
(24)            $Q = 1;$ 
(25)         else if  $A(i, j) == 255$  then
(26)           Determine whether there is  $T$  before 0 is encountered in the four directions of point  $A(i, j)$ .
(27)        $T = T + 1$ 
(28)   for  $i = 1; i \leq T; i++$  do
(29)     if there is  $i$  on the boundary or the total number of pixels with a value of  $i$  is less than 300  $\triangleright$  These two cases are not considered as
a room
(30)     then
(31)       continue;
(32)     else
(33)       Convert all  $i$  to 0 and convert other values to 255.
(34)     Stack the decor items on the respective detected rooms.

```

ALGORITHM 2: Room segmentation and decor items retrieval.

pixel is set to the value of $T = T + 1$, which is 2. Hence, it will be treated as the reference point again to update the surrounding pixels with value $T + 1$. The above-mentioned steps are repeated until pixel intensity 255 will no longer be found. Intuitively, the newly found pixel with the intensity of 255 is regarded as the new room definition. As such, the resultant T value denotes the total number of the new room formed in an image. Then, lines 28 to 33 determine if the room satisfies the room size requirement. Particularly, if the room has an area size of fewer than 300 pixels, the room is ignored. This is because there are some unusual rooms as the images are artificially generated by the computer. Finally, line 34 stacks the decor items to the individual detected rooms.

3.4. Bedroom Classification. This stage differentiates the rooms detected into bedroom/nonbedroom categories. The convolutional neural network (CNN) is employed as both the feature extractor and classifier for the bedroom classification. Intuitively, the shape, characteristics, patterns of a bed are the key features to decide if the image is a bedroom/nonbedroom. Thus, CNN architectures are expected to learn the features of the bed in order to make the correct predictions.

Several pretrained neural networks (i.e., AlexNet [22], ResNet [23], SqueezeNet [24], and GoogLeNet [25]) are utilized with slight modification. Note that, most of the parameters existing in the networks are transferred to adapt and learn the new characteristics of the floor plan images. Specifically, the parameters will be fine-tuned automatically throughout the learning progress. Concretely, these network architectures are comprised of five types of operation: convolution, ReLU, pooling, fully connected, and dropout. The bedroom images are first standardized to a certain size (i.e., $\aleph \times \aleph$).

- (1) Convolution and ReLU: The image performs a dot product between a kernel/weight and the local regions of the image. This step can achieve blurring, sharpening, edge detection, and noise reduction effect. ReLU is an elementwise activation function and is usually applied as the thresholding technique to eliminate the neurons that are playing a vital role in discriminating the input and is essentially meaningless.

Each e_{ij} pixel in the image is defined as

$$e_{ij}^l = \{f^l(x_{ij}^l + b^l) | i = 1, 2, \dots, \aleph, j = 1, \dots, \aleph\},$$

where $x_{ij}^l = \sum_{a=0}^{m-1} \sum_{b=0}^{m-1} w_{ab}^l y_{(i+a)(j+b)}^{l-1}$,

(4)

l indicates the layer, x_{ij}^l is the pixel-value vector corresponding to e_{ij} pixel, f^l is the ReLU activation function, w^l is the coefficient vector, and b^l is the bias determined by the feature map.

Thus, for an input x , the ReLU function can be indicated as

$$f(x) = \max(0, x). \quad (5)$$

- (2) Pooling: To downsample the image along the spatial dimensions (i.e., width and height). This allows dimension reduction and enables the computation process to be less intensive.

The k -th unit of the image feature in the pooling layer is expressed as

$$\text{Pool}_k = f(\text{down}(C) * W + b), \quad (6)$$

where W and b are the coefficient and bias, respectively. $\text{down}(\cdot)$ is a subsampling function:

$$\text{down}(C) = \max\{C_{s,l} | s \in Z^+, l \in Z^+ \leq m\}. \quad (7)$$

$C_{s,l}$ is the pixel value of C in e and m indicates the sampling size.

- (3) Fully connected: All the previous layer and the next layer of neurons are linked. It acts like a classifier based on the features from the previous layer.
- (4) Dropout: The neurons are randomly dropped out during the training phase. This can avoid overfitting phenomena and enhance the generalization of the neural network trained.

Figure 2(f) shows the categorization of each partitioned room to bedroom or nonbedroom after adopting the CNN method.

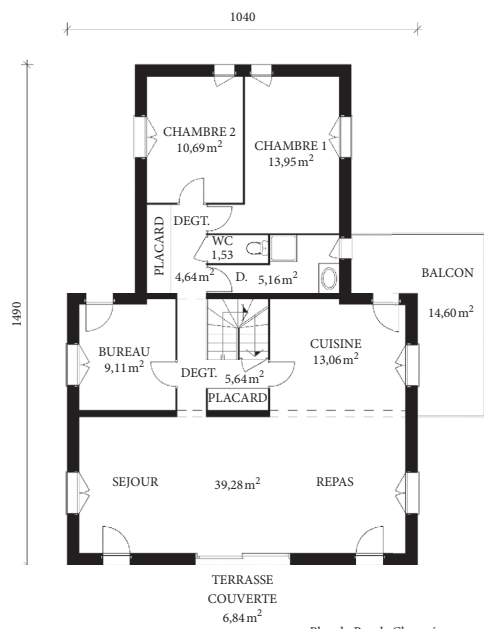
4. Dataset Description

Thus far, there are limited publicly available datasets that contain the architectural floor plan images. For instance, these four datasets: CVC-FP [26], SESYD [27], Robin [28], and Rent3D [29] usually served as the experimental data in academic studies. The detailed information of these datasets is shown in Table 1 and the sample images are illustrated in Figure 4. It is observed that the databases are largely limited on their own. Concisely, the floor plan images may vary in different aspects: (1) building types (i.e., theater, school, house, and museum); (2) multidimensional images (i.e., 2D and 3D); (3) representation types (i.e., sketches and computer-aided design); and (4) furniture layouts (i.e., walls, windows, doors, sofa, and stairs). Particularly, Rent3D and CVC-FP are the scanned images. The contents are mostly in text, rather than displaying the furniture icon. On the other hand, Robin and SESYD comprised the computer-generated floor plan with lesser image noise, compared to the other two datasets. However, the wall thickness of SESYD is relatively thin. Therefore, it may lead to some errors during the wall segmentation process. Based on the pros and cons of the datasets discussed above, only the Robin dataset is considered as the experimental data in this paper.

From the pros and cons summarized in Table 1, we opt to select the Robin dataset as the experiment data to evaluate the proposed framework. In short, the repository of the building plans (Robin) dataset can be categorized into three main classes, namely, 3 rooms, 4 rooms, and 5 rooms. The

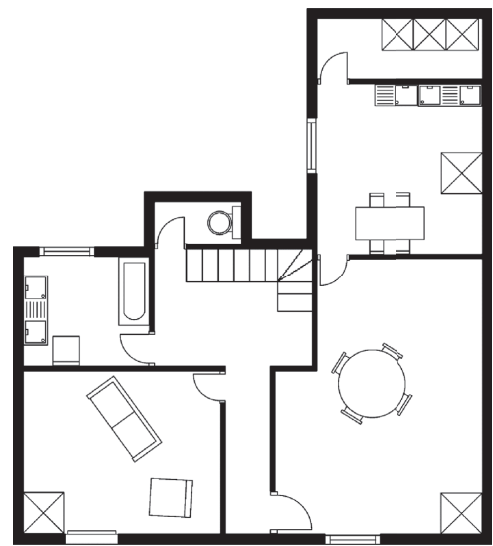
TABLE 1: Detailed information of the four floor plan databases.

	CVC-FP	SESYD	Robin	Rent3D
Presence of text	✓	✗	✗	✓
Total number of images	122	10	510	250
Item	Table	✗	✓	✗
	Chair	✗	✓	✗
	Sink	✓	✓	✓
	Toilet bowl	✓	✓	✓
	Bathtub	✓	✓	✓
	Bed	✗	✓	✓
	Cabinet	✗	✓	✗
	Sofa	✗	✓	✗
Fake/generated	✗	✓	✓	✗
Real/scan	✓	✓	✗	✓

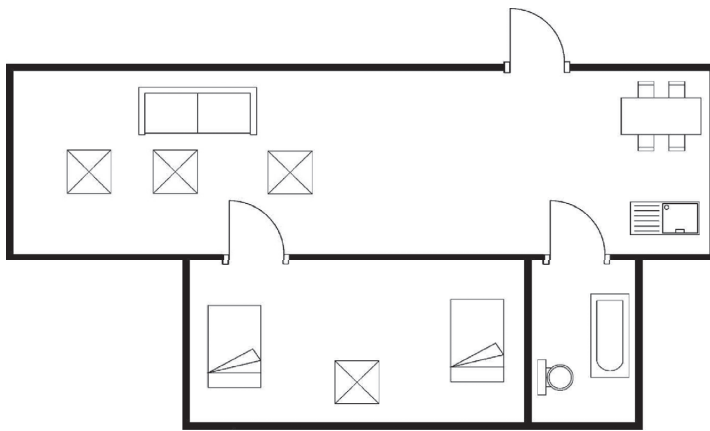


Plan du Rez de Chaussée
échelle 1/100é soit 1 cm pour 1 m
alain meunier architecte d.p.l.g.

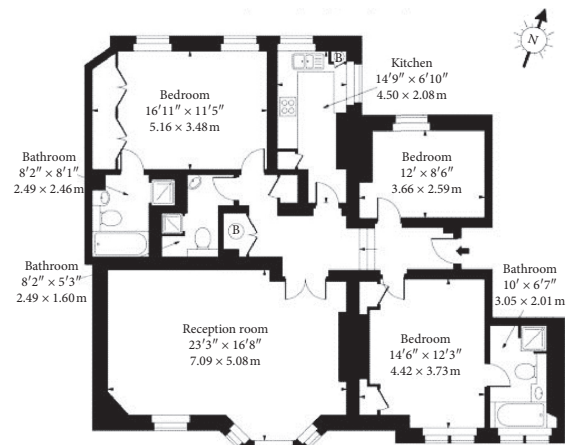
(a)



(b)



(c)



(d)

FIGURE 4: Sample images from the four datasets: (a) CVC-FP, (b) SESYD, (c) Robin, and (d) Rent3D.

images for the three classes compose an equal number of the sample which are 170 each. The item contained in the images includes table, chair, sink, toilet bowl, bathtub, bed, and sofa. The spatial resolutions for the images are varied, with the average sizes of 1085×1115 , 1188×1280 , and 1593×1859 for 3 rooms, 4 rooms, and 5 rooms, respectively. The dataset contains both the grayscale and colored images. However, all the images are portrayed in monochromatic color, in which only actual white and black pixels are presented. The illustration of the sample images and the details of each image category are portrayed in Table 2. Note that the images shown are randomly selected from the dataset. The image data that support the findings of this study are openly available in [30].

Nonetheless, there is no textual description provided for this floor plan dataset. Since this dataset is synthetic data and the elements are aggregated automatically, some image samples may not be in accordance with the real-world scenario. For example, it can be seen that the sample image #1 looks more normal compared to that of the sample image #2. For example, the sample image #2 for the 3 rooms does not have any object/graphics in one of the rooms. As for the sample image #2 for the 4 rooms, three rooms are encircled by a big squared room. Lastly, for the sample image #2 for the 5 rooms, to enter one of the bedrooms, it has to go through another bedroom. Besides, the only route to the kitchen is to pass through a bathroom.

Surprisingly, among the 510 images in the Robin dataset, 22 images do not contain any bedroom furniture. Thus, the 22 images are regarded as a 0-bedroom category. Most of the images comprise one bedroom, namely, 391 images. The maximum number of bedrooms in the dataset is two, in which up to 97 images contain two bedrooms. An overview regarding the number of the bedroom with their respective sample images is illustrated in Table 3.

5. Experiment Results and Discussion

5.1. Performance Metric. There are two evaluation metrics to validate the performance of the proposed framework, namely, accuracy and $F1$ -score [31]. Specifically, accuracy is the most intuitive performance measure and shows how accurate and precise the result is generated. Since the number of bedrooms is inconsistent in the Robin dataset, $F1$ -score is used to tackle to imbalance class problem and to avoid the bias phenomena. Mathematically, these two metrics can be expressed as

$$Accuracy := \frac{TP + TN}{TP + FP + TN + FN}, \quad (8)$$

and

$$F1 - score := 2 \times \frac{Precision \times Recall}{Precision + Recall}, \quad (9)$$

where

$$Recall := \frac{TP}{TP + FN}, \quad (10)$$

and

$$Precision := \frac{TP}{TP + FP}, \quad (11)$$

where

- (1) TP (true positive): the model correctly classified the bedroom
- (2) TN (true negative): the outcome where the model correctly predicts that it is not a bedroom
- (3) FN (false negative): the event where the model does not predict the bedroom correctly, while in fact, it is a bedroom
- (4) FP (false positive): the test result indicates that it is a bedroom, but it is not

5.2. Classification Performance and Analysis. All the 510 floor plan images from the Robin dataset are used as the experimental data. The bedroom classification results using five different CNN architectures (i.e., AlexNet, GoogleNet, SqueezeNet, ResNet-101, and ResNet-50) are shown in Table 4. Note that the number of epochs for each CNN is set to the range of 10 to 100. The learning rates and the minibatch size are set to 0.0001 and 128. The details of the configuration of the CNN architecture and parameter values are listed in Table 5.

The three types of images are merged into a composite dataset to provide a fair training and testing experiment environment. Concisely, in the classification stage, 5-fold cross-validation is utilized to validate the performance of the transfer learning model on an unseen image sample. Thus, for each fold, $510/5 = 102$ images are treated as the testing sample, while the remaining 408 images served as the input to the architecture for parameter training (i.e., weights and biases). This process will be repeated 5 times until all the images have been tested at least one time. Note that, there is no overlapping data between the training and testing sets.

Table 4 reports one of the highest classification performances obtained among the epoch range. It is observed that the highest accuracy exhibited is produced by GoogleLeNet, in which the accuracy and $F1$ -score obtained are 98.40% and 98.80%, respectively. From Table 4, it can be seen that although ResNets are the two largest architectures among all the CNN methods, the classification results are the lowest (i.e., accuracy = 81.40% and $F1$ -score = 88%). This implies that learning the bedroom features leads to an overfitting phenomenon when a huge CNN is employed. Nevertheless, this binary classification task with limited data samples has demonstrated that the transfer learning technique is capable of training the discriminant features on small-size data.

To provide further insights into the classification performance, a detailed investigation is done regarding some correct and incorrect predicted bedroom images. Since there are a total of 510 images in the dataset, Table 6 displays partial numerical results that are randomly selected from the dataset. Among the samples, 8 cases produce the $F1$ -score of 100%, whereas 2 cases generate 66.67%. The reason for false detection may be because of

TABLE 2: Detailed information of Robin dataset [28].

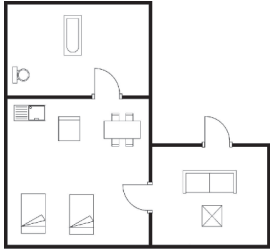
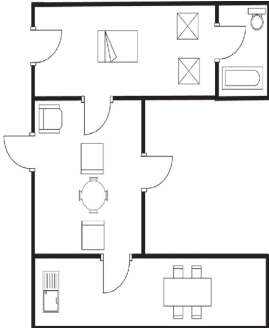
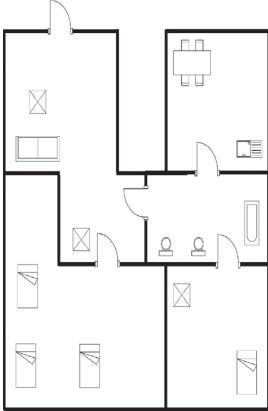

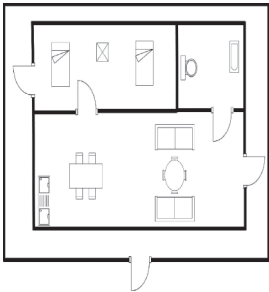
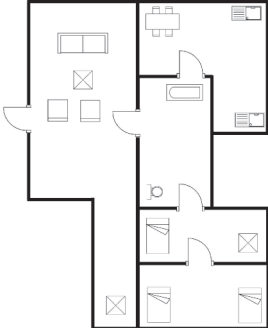
	3 rooms	4 rooms	5 rooms	
Image number	170	170	170	
Resolution	(Average) (Minimum) (Maximum)	1085 × 1115 603 × 1194 1204 × 1244	1188 × 1280 793 × 1228 1444 × 2050	1593 × 1859 1116 × 2365 2404 × 2380
Sample picture (#1)				
Sample picture (#2)				

TABLE 3: The number of bedroom in the Robin dataset [28].

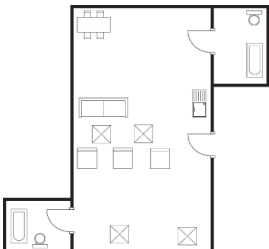
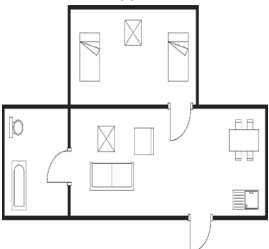
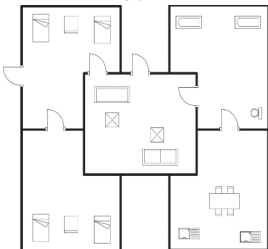
	0 bedrooms	1 bedroom	2 bedrooms
Image number	22	391	97
Sample picture			

TABLE 4: Performance % of the bedroom classification when utilizing five convolutional neural networks, in terms of accuracy (Acc) and F1-score (F1)

Method	Epoch	Acc	F1
AlexNet	60	90.5	97.5
GoogLeNet	30	98.4	98.8
ResNet-50	60	81.8	88
ResNet-101	30	81.8	87.2
SqueezeNet	70	96.6	97.9

TABLE 5: Configuration type and parameters of CNN models.

Configuration type	Parameter
Optimizer	SGDM
Momentum	0.9000
InitialLearnRate	0.0001
LearnRateSchedule	Piecewise
LearnRateDropFactor	0.2000
LearnRateDropPeriod	10
L2Regularization	0.0001
GradientThresholdMethod	L2 norm
GradientThreshold	Inf
MaxEpochs	[10, 100]
MiniBatchSize	128
Verbose	1
VerboseFrequency	50
ValidationPatience	Inf
Shuffle	Once
ExecutionEnvironment	Auto

TABLE 6: Detailed analysis of ten floor plan samples with GoogLeNet.

Image	TP	TN	FP	FN	F1-score (%)
Rob_001	2	1	0	0	100
Rob_002	1	1	0	1	66.67
Rob_003	1	1	0	1	66.67
Rob_004	2	1	0	0	100
Rob_005	2	1	0	0	100
Rob_006	2	1	0	0	100
Rob_007	2	1	0	0	100
Rob_008	2	1	0	0	100
Rob_009	3	1	0	0	100
Rob_010	2	1	0	0	100

several factors. For instance, the insufficient training sample makes the architecture not trained well. Besides, an unusual room image is shown in sample picture #2 for 4 rooms in Table 2. Nevertheless, the accuracy and *F1*-score attained are satisfactory when applying some popular CNN architecture, such as GoogLeNet.

Aside from the numerical results, the qualitative visualization is shown in Figures 5 and 6 to provide further classification context clues. Figure 5 depicts the activations of GoogLeNet. This particular activation layer selected is allocated in the third quarter of the network. There are noticeable brighter intensity pixels when there is a bed at the respective position. Besides, the activations of ResNet for the layer at a similar location (i.e., the third quarter of the network) are shown in Figure 6 for a fair comparison. Apparently, Figure 6(a) denotes that the bottom left corner should have a bedroom, as this particular in the activation image has bright pixels. However, that location does not have any bedroom, which means that the bedroom features learned in this network are insufficiently precise.

The detailed properties of all the network architectures are tabulated in Table 7, which include the depth, size, and the number of the learnable parameter in the network. Amongst, SqueezeNet occupies the least size (5 MB) and the

fewest parameters (1 million), but the network depth is ~ 2.5 times larger than the AlexNet. The deepest network is ResNet-101, which has a depth of 347. As the largest network that contains the largest number of the learnable parameter is AlexNet size of 61 million, AlexNet is the shallowest network among the networks presented (depth size of 25).

The proposed method is capable of tackling the images with different orientations as the transfer learning architecture is designed such that the model is invariant to orientation. Besides, the architecture learns the discriminative features automatically from the floor plan images while achieving high performance. Moreover, with a limited training sample, the classification performance is promising, namely, *F1*-score = 98.8% when employing GoogLeNet. On the other hand, the epoch number required for model training is relatively few (< 100), implying that the speed of computing and execution is quick. This primarily contributed to the advantages of the transfer learning technique, as the architecture was pretrained in an ImageNet database that consists of a million images with 1000 categories.

Since GoogLeNet exhibits the best results when performing the binary classification task. We attempt to extend the work by summarizing the number of correctly guessing images. Note that, each floor plan image may contain up to 5 rooms, yet, the bedroom numbers ranged from 0 to 2. Among the 510 images, 457 images are correctly classified to their respective bedroom number. A confusion matrix is tabulated in Table 8, providing a detailed analysis regarding the classification task. It is discerned that all the 22 images that do not have bedrooms are correctly classified. However, it is also noticed that 4 images that are supposed to contain only 1 or 2 bedrooms are classified as having 3 bedrooms. Nevertheless, the proposed algorithm exhibits a promising bedroom classification of 89.61%. Moreover, the gradient-weighted class activation mapping (Grad-CAM) [32] method is utilized to highlight important regions in images that are trained by the CNN model. The interpretation of GoogLeNet model predictions using Grad-CAM is portrayed in Figure 7. These pleasing numerical and qualitative performances further verify the capability and feasibility of the framework developed.

To further verify the effectiveness of the proposed framework, the performance compared to the state of the art is tabulated in Table 9 when conducting different detection tasks on the floor plan images. Succinctly, Table 9 lists the accuracy of (a) bedroom detection, the detected rooms are correctly recognized as the bedroom; (b) room detection, the rooms are successfully being detected; and (c) room-type detection, the detected rooms are correctly classified as the room-type, such as bedroom, drawing room, and kitchen. On the other hand, nonroom-type include the parking porch, bathroom, study room, and prayer room. The overall accuracy achieved in the previously published work is quite promising, namely, $> 80\%$. Note that, methods #1-#2 evaluated the algorithms on the R2V dataset [33], method #3 evaluated on Rent3D dataset [29], and methods #4-#7 evaluated on the CVC-FP dataset [26]. The details of the datasets are summarized in Table 1. Notably, the proposed method herein is capable of



FIGURE 5: Activations of GoogLeNet when the image (a) does not contain a bed and (b) contains a bed.

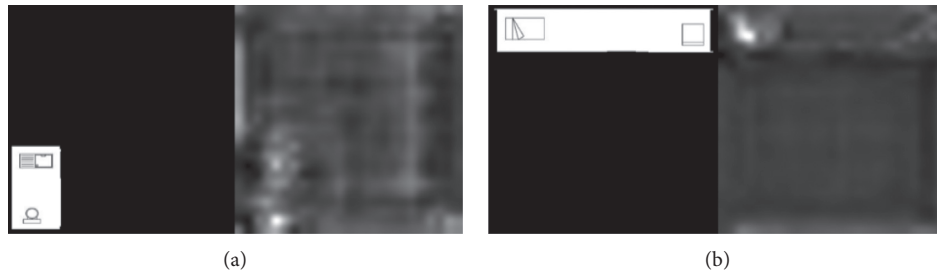


FIGURE 6: Activations of ResNet when the image (a) does not contain a bed and (b) contains a bed.

TABLE 7: The properties of the network architecture.

Method	Depth	Size (MB)	Parameter (millions)
AlexNet	25	244.66	60.95
GoogLeNet	144	29.06	6.99
ResNet-50	177	103.68	25.50
ResNet-101	347	180.34	44.44
SqueezeNet	68	5.23	1.23

TABLE 8: Confusion matrix of bedroom number calculation when utilizing GoogLeNet as the feature descriptor.

		Predicted			
		0	1	2	3
Desired	0	22	0	0	0
	1	20	355	14	2
	2	0	15	80	2
	3	0	0	0	0

detecting all the rooms and exhibiting an accuracy of 98% when detecting the bedroom.

Moreover, the recommender system can provide additional details regarding the floor plan information. In summary, a list of statistics can be generated, such as (1) the total built-up area of the floor plan; (2) the total number of bedroom and their area; (3) the total number of nonbedroom and their area; (4) the position of the bedrooms; (5) the

maximum and minimum area of each room; (6) the total number of doors; and (7) the total number of doors in each room. Taking Figure 7(a) as an example, the list of detailed statistics is tabulated in Table 10. Note that, the original floor plan in the dataset did not provide the real-world scale. Thus, the unit of the built-up area presented here is pixels². The unit (i.e., pixels²) can be easily converted to the real-world measurement (i.e., square feet, m²) by a ratio proportion.

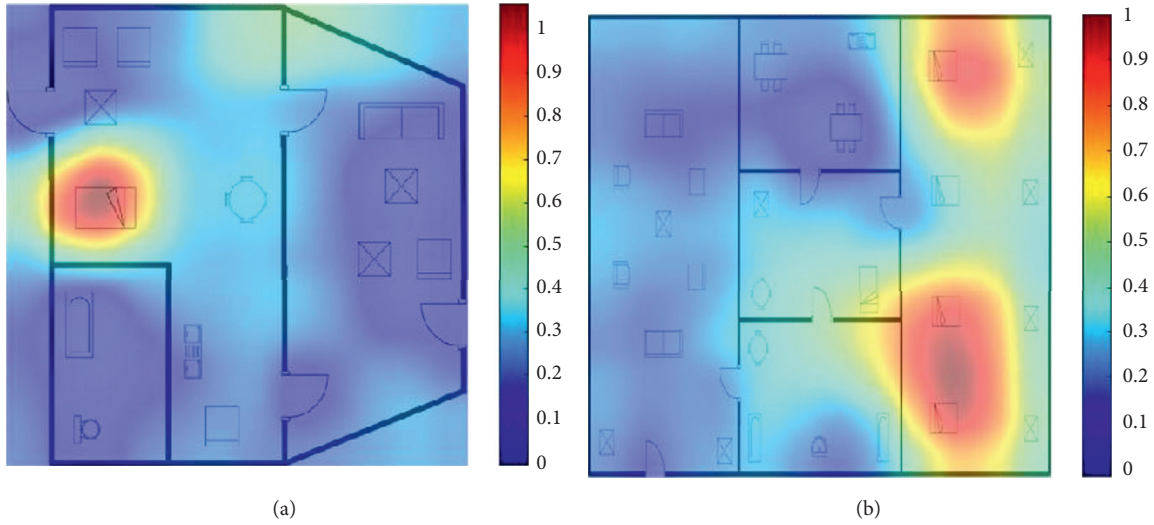


FIGURE 7: The Grad-CAM visualization to localize class-discriminative regions.

TABLE 9: Performance comparison of the bedroom detection and room detection.

No.	Method	Bedroom detection	Room detection	Room-type detection
1	[33]	0.89	—	—
2	[34]	0.83	—	—
3	[34]	0.75	—	—
4	[21]	—	0.85	—
5	[17]	—	0.89	—
6	[13]	—	0.91	—
7	[35]	—	0.86	0.94
8	Ours	0.98	1	—

TABLE 10: List of statistics generated by the recommender system.

Statistics	Values
Total built-up area	118,278 pixels ²
Total number of bedrooms	1
Bedroom's area	56,765 pixels ²
Total number of nonbedrooms	2
Nonbedroom's area	[46629, 14884] pixels ²
Position of the bed	West
Maximum area of room	56,765 pixels ²
Minimum area of room	14884 pixels ²
Total number of doors	4
Number of doors in each room	[0, 3, 3]

6. Conclusion

This paper presents a novel framework to automatically identify the location and the number of bedrooms from the floor plan images. Some traditional and data-driven image processing techniques are applied. In brief, Otsu's thresholding and morphological operations are employed to preprocess the image. Then, the rooms are extracted using the Hough transform and FAST algorithm. Finally, some popular convolutional neural network architectures are utilized to determine if the detected room is the bedroom. The quantitative performance results suggest that the proposed pipeline is feasible in recommending the house property from

architectural floor plan images. Particularly, an excellent bedroom classification accuracy of 98.4% and *F1*-score of 98.8% are achieved when employing the state-of-the-art deep learning techniques. Moreover, the visual presentation regarding the cues of the correctly detected bedroom category further verifies the effectiveness of the approach.

As future work, this framework can be extended to recognize other function rooms, such as the bathroom, dining room, or living room. Besides, the binary classification task and the limited data samples point towards the studies on the new design of the shallow neural network. Note that, some previously published works investigated the floor plan with both the text and graphics. However, the experiments presented in this work did not consider the images with text, and this points towards an important direction for future research. Therefore, it is also worth investigating the optical character recognition (OCR) technique on other types of floor plan images.

Data Availability

The image data that support the findings of this study are openly available in <https://github.com/gesstalt/ROBIN>.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was funded by Ministry of Science and Technology (MOST) (Grant nos. MOST 109-2221-E-035-065-MY2, MOST 108-2218-E-035-018-, and MOST 108-2218-E-009-054-MY2).

References

- [1] F. Isensee, P. F. Jaeger, S. A. A. Kohl, J. Petersen, and K. H. Maier-Hein, "Nnu-net: a self-configuring method for deep learning-based biomedical image segmentation," *Nature Methods*, vol. 18, no. 2, pp. 203–211, 2021.
- [2] H. Lee, H. Ho, and Y. Zhou, "Deep learning-based monocular obstacle avoidance for unmanned aerial vehicle navigation in tree plantations," *Journal of Intelligent and Robotic Systems*, vol. 101, no. 1, pp. 1–18, 2021.
- [3] K.-H. Liu, Q.-S. Jin, H.-C. Xu, Y.-S. Gan, and S.-T. Liong, "Micro-expression recognition using advanced genetic algorithm," *Signal Processing: Image Communication*, vol. 93, Article ID 116153, 2021.
- [4] E. V. Lavrukhin, K. M. Gerke, K. A. Romanenko, K. N. Abrosimov, and M. V. Karsanina, "Assessing the fidelity of neural network-based segmentation of soil xct images based on pore-scale modelling of saturated flow properties," *Soil and Tillage Research*, vol. 209, Article ID 104942, 2021.
- [5] Williamsburg, "Why are floor plans not used in real estate marketing?" 2020, <https://mrwilliamsburg.com/why-are-floor-plans-not-used-in-real-estate-marketing/>.
- [6] J. Bayer, S. S. Bukhari, and A. Dengel, "Interactive design support for architecture projects during early phases based on recurrent neural networks," in *Proceedings of the International Conference on Pattern Recognition Applications and Methods*, pp. 27–43, Springer, Funchal, Madeira, Portuga, January 2018.
- [7] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [8] G. Egor, S. Sven, D. Martin, and K. Reinhard, "Computer-aided approach to public buildings floor plan generation. magnetizing floor plan generator," *Procedia Manufacturing*, vol. 44, pp. 132–139, 2020.
- [9] C. Liu, J. Wu, and Y. Furukawa, "Floornet: a unified framework for floorplan reconstruction from 3d scans," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 201–217, Munich, Germany, September 2018.
- [10] A. Phalak, V. Badrinarayanan, and A. Rabinovich, "Scan2-plan: efficient floorplan generation from 3d scans of indoor scenes," arXiv preprint arXiv:2003.07356, 2020.
- [11] J. Zheng, J. Zhang, J. Li, R. Tang, S. Gao, and Z. Zhou, "Structured3d: a large photo-realistic dataset for structured 3d modeling," vol. 2, no. 7, arXiv preprint arXiv:1908.00222, 2019.
- [12] J. Chen, C. Liu, J. Wu, and Y. Furukawa, "Floor-sp: inverse cad for floorplans by sequential room-wise shortest path," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 2661–2670, Seoul, Korea, October 2019.
- [13] L.-P. de las Heras, S. Ahmed, M. Liwicki, E. Valveny, and G. Sánchez, "Statistical segmentation and structural recognition for floor plan interpretation," *International Journal on Document Analysis and Recognition*, vol. 17, no. 3, pp. 221–237, 2014.
- [14] R. Khade, K. Jariwala, C. Chattopadhyay, and U. Pal, "A rotation and scale invariant approach for multi-oriented floor plan image retrieval," *Pattern Recognition Letters*, vol. 145, pp. 1–7, 2021.
- [15] S. Dodge, J. Xu, and B. Stenger, "Parsing floor plan images," in *Proceedings of the 2017 Fifteenth IAPR International Conference on Machine Vision Applications (MVA)*, pp. 358–361, IEEE, Nagoya, Japan, May 2017.
- [16] S. Ahmed, M. Liwicki, M. Weber, and A. Dengel, "Improved automatic analysis of architectural floor plans," in *Proceedings of the 2011 International Conference on Document Analysis and Recognition*, pp. 864–869, IEEE, Washington, DC, USA, September 2011.
- [17] S. Ahmed, M. Liwicki, M. Weber, and A. Dengel, "Automatic room detection and room labeling from architectural floor plans," in *Proceedings of the 2012 10th IAPR International Workshop on Document Analysis Systems*, pp. 339–343, IEEE, Washington, DC, USA, March 2012.
- [18] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: speeded up robust features," *Computer Vision - ECCV 2006*, Springer, in *Proceedings of the European conference on computer vision*, pp. 404–417, Springer, Graz, Austria, May 2006.
- [19] S. Ahmed, M. Weber, M. Liwicki, and A. Dengel, "Text/graphics segmentation in architectural floor plans," in *Proceedings of the 2011 International Conference on Document Analysis and Recognition*, pp. 734–738, IEEE, Washington, DC, USA, September 2011.
- [20] S. Goyal, S. Bhavsar, S. Patel, C. Chattopadhyay, and G. Bhatnagar, "Sugaman: describing floor plans for visually impaired by annotation learning and proximity-based grammar," *IET Image Processing*, vol. 13, no. 13, pp. 2623–2635, 2019.
- [21] S. Macé, H. Locteau, E. Valveny, and S. Tabbone, "A system to detect rooms in architectural floor plan images," in *Proceedings of the 9th IAPR International Workshop on Document Analysis Systems*, pp. 167–174, ACM, New York, NY, USA, June 2010.
- [22] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proceedings of the Advances in neural information processing systems*, pp. 1097–1105, Lake Tahoe, Nevada, December 2012.
- [23] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, Las Vegas, NV, USA, June 2016.
- [24] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "Squeezenet: alexnet-level accuracy with 50x fewer parameters and < 0.5 mb model size," arXiv preprint arXiv:1602.07360, 2016.
- [25] C. Szegedy, W. Liu, Y. Jia et al., "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9, Boston, MA, USA, June 2015.
- [26] L.-P. de las Heras, O. R. Terrades, S. Robles, and G. Sánchez, "Cvc-fp and sgt: a new database for structural floor plan analysis and its groundtruthing tool," *International Journal on Document Analysis and Recognition*, vol. 18, no. 1, pp. 15–30, 2015.
- [27] M. Delalandre, E. Valveny, and J. Y. Ramel, "Recent contributions on the sesyd dataset for performance evaluation of symbol spotting systems," Retrieved from semanticscholar.org, Dec. 2017. 2011, <https://www.semanticscholar.org/paper/Recent-contributions-on-the-SESYD-dataset-for-of-Delalandre-Valveny/42a3d89544393fe80acb6d6c4eae0239c9c96b99>.
- [28] D. Sharma, N. Gupta, C. Chattopadhyay, and S. Mehta, "Daniel: a deep architecture for automatic analysis and retrieval of building floor plans," in *Proceedings of the 2017 14th IAPR International Conference on Document Analysis and*

- Recognition (ICDAR)*, vol. 1, pp. 420–425, IEEE, Kyoto, Japan, November 2017.
- [29] C. Liu, A. G. Schwing, K. Kundu, R. Urtasun, and S. Fidler, “Rent3d: floor-plan priors for monocular layout estimation,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3413–3421, Boston, MA, USA, June 2015.
- [30] Robin (repository of building plans): <https://github.com/gesstalt/ROBIN>.
- [31] M. Sokolova, N. Japkowicz, and S. Szpakowicz, “Beyond accuracy, f-score and roc: a family of discriminant measures for performance evaluation,” in *Proceedings of the Australasian joint conference on artificial intelligence*, pp. 1015–1021, Lecture Notes in Computer Science, Springer, Hobart, Australia, December 2006.
- [32] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, “Grad-cam: visual explanations from deep networks via gradient-based localization,” in *Proceedings of the IEEE international conference on computer vision*, pp. 618–626, Venice, Italy, October 2017.
- [33] C. Liu, J. Wu, P. Kohli, and Y. Furukawa, “Raster-to-vector: revisiting floorplan transformation,” in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2195–2203, Venice, Italy, October 2017.
- [34] Z. Zeng, X. Li, Y. K. Yu, and C. W. Fu, “Deep floor plan recognition using a multi-task network with room-boundary-guided attention,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9096–9104, Seoul, Korea, October 2019.
- [35] H. K. Mewada, A. V. Patel, J. Chaudhari, K. Mahant, and A. Vala, “Automatic room information retrieval and classification from floor plan using linear regression model,” *International Journal on Document Analysis and Recognition*, vol. 23, no. 4, pp. 253–266, 2020.

Review Article

Improved AND/OR Tree Search Algorithm in Analysis of Stochastic and Time-Dependent Shortest Path Problem

Zhi-ying Xie ^{1,2}, Yuan-Rong He ^{1,2}, Yuan-tong Jiang^{3,4} and Chih-Cheng Chen ^{5,6}

¹School of Computer and Information Engineering, Xiamen University of Technology, Xiamen 361024, China

²Big Data Institute of Natural Hazards Monitoring for Digital Fujian, Xiamen, Fujian 361024, China

³School of Cultural Industries and Tourism, Xiamen University of Technology, Xiamen 361024, China

⁴Research Center of Cultural Industries, Fujian Social Science Research Base, Xiamen 361024, China

⁵School of Ocean Information Engineering, Jimei University, Xiamen 361021, China

⁶Department of Aeronautical Engineering, Chaoyang University of Technology, Taichung 413, Taiwan

Correspondence should be addressed to Yuan-Rong He; 2012112001@xmut.edu.cn and Chih-Cheng Chen; ccc@gm.cyut.edu.tw

Received 5 April 2021; Revised 9 June 2021; Accepted 14 July 2021; Published 28 July 2021

Academic Editor: Qianchuan Zhao

Copyright © 2021 Zhi-ying Xie et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Real-time vehicle guidance effectively reduces traffic jams and improves the operational efficiency of urban transportation. The trip time on a route is considered as a random process that changes with time, and the shortest path selection requires a random dynamic model and the solution of a decision-making problem. Thus, the shortest trip time is the criterion to determine the dynamic path selection by a random dynamic programming (DP) model which discretizes the trip times in the continuous segments on the route. In this study, a numerical model of random dynamic programming is established by using a probability tree model and an AND/OR (AO*) algorithm to select the path of the shortest trip time. The results show that the branches of the probability tree are only accumulated on the “quantity” and do not cause a “qualitative” change. The inefficient accumulation of “quantity” affects the efficiency of the algorithm, so it is important to separate the accumulation of “quantity” from node expansion. The accumulation of “quantity” changes the trip time according to the entering time into a segment, which demands an improved AO* algorithm. The new AO* algorithm balances between efficiency and the trip time and provides the optimal real-time vehicle guidance on the road.

1. Introduction

Vehicle-to-everything (V2X) and Internet of Things (IoT) collect a large amount of observation data of traffic from multisource sensors and devices and require big data technology [1–4]. The collected data may be used to provide the optimal path for vehicles, especially unmanned vehicles. In unmanned driving, dynamic path planning is required in two aspects: global path planning [5] and obstacle avoidance [6]. The algorithms of obstacle avoidance in path planning [7–9] and global path planning [10–12] are attracting considerable interest. However, the randomness of the traffic data from the variable traffic network environment complicates the algorithms, which still need more research than before.

The trip time in a segment of a vehicle path is regarded as a time-dependent random variable, and its probability distribution depends on the entering time of a vehicle to the road segment [13]. Thus, the problem to obtain the shortest time of the vehicle’s trip in the segment is the shortest path problem of a stochastic time-dependence (STD) network. Hall [14] stated that a label setting (LS) or a label correcting (LC) algorithm was not suitable for the shortest path as Bellman’s optimality principle does not hold randomness [15].

The optimal path selection is not a simple shortest path problem, but an adaptive decision-making problem with time. The adaptive decision-making problem cannot be solved by a fixed shortest path method. The optimal path selection strategy from a starting to the next target node

considers the time to move between them. The path selection is not only needed from the starting node but from each decision point (intersection) to reach the target node in the shortest time. The optimal path must be selected to have the shortest trip time to the target node or intersection, and the subsequent selection will be no longer affected by the previous selection process. For this type of decision-making, dynamic programming is regarded to be appropriate. Thus, a problem of stochastic and time-dependent shortest path (STDSP) requires a model of dynamic programming.

This study aims to suggest an AND/OR (AO*) algorithm to solve an STDSP problem in finding the optimal path with the shortest trip time between nodes or intersections. The complexity of the shortest path problem in an STD network was analyzed by using a probability tree diagram and a first in first out (FIFO) algorithm based on the nonsatisfaction and nonuniqueness of the trip times. Then, a model of stochastic dynamic programming for dynamic path selection was applied to the estimation of the minimum trip times. Based on the heuristic function and the two-point discretization of the trip time between segments (between nodes or intersections), an AO* algorithm, a heuristic search algorithm was proposed to solve the STDSP problem by using the STD programming model with the consideration of the balances between the efficiency and the optimal trip. With the results, the STDSP problem was defined with the analysis of the complexity of the STDSP problem. Then, the algorithm was validated with real-time observation data. Finally, the proposed AO* algorithm was improved again for suggesting the final algorithm.

2. Theoretical Background

Bellman's optimality principle is used to find the shortest path between any of the two nodes in a network and applied to a static shortest path algorithm. When the trip time between the nodes is constant, that is, the path is time-independent, an efficient labeling algorithm solves the shortest path problem. Wu et al. demonstrated [16] that the Bellman optimal principle in the static network is also applicable to the STD and FIFO networks. In the FIFO network, the algorithm of the shortest path in the static network can be used for solving the shortest path in a dynamic network. Thus, LS or LC algorithms solve such shortest path problems. When a trip time is regarded as a random variable with a known probability distribution with the objective function, the random shortest path problem is transformed into a deterministic one for obtaining a desired shortest path. However, a random trip time can be obtained by an efficient labeling algorithm by solving the expected shortest path problem [17]. The shortest time is obtained as the solution of an objective function of the path between nodes, which does not consider the random characteristics of the network [18].

In the actual traffic, the minimum trip time, as well as the minimum risk, is considered [19]. Therefore, the minimum expectation-mean square error [20] and α -reliable path problem need to be considered [21]. The path objective function of the minimum expectation-mean-square error path is a linear combination of the expected value and the

mean square error. Thus, the proportional coefficient of the linear combination needs an artificial setting that is arbitrary and cannot uniformly explain the path selection.

Dynamic programming is appropriate for multistage decision-making problems. Steinmetz et al. [22] modeled a dynamic path selection as a Markov decision process (MDP) and believed that each segment on the road corresponded to blocking and nonblocking. When the current state of each segment is known, an intersection to pass is selected. Hall [14] suggested the use of dynamic programming to solve the time-dependent path selection and proposed the algorithm that combined a k-shortest path and a branch and bound method. This algorithm had low operating efficiency and could not satisfy the requirement in real time. When Wu et al. [16] summarized the rules for solving STDP problems using the LC algorithm which had randomness. They believed that replacing the definite value with the expected value of a random variable allowed similar results to the definite situation by assuming consistency. However, the assumption is not applicable in the actual transportation network as the real situation has randomness.

Fu and Rilett [23] regarded a random time-dependence problem of the path selection by Hall [14] to be a continuous random process. They transformed the time-dependence problem into an extreme value problem to estimate the trip time on the path. Based on the trip time and its variance in each segment, a series of probability-based approximation models (PAMs) were established by using the k-shortest path. Then, a heuristic algorithm was used to solve the STDSP problem. Fu [24] proposed an adaptive strategy and used dynamic programming for the PAM. However, his adaptive LC algorithm did not adopt the nondifferentiability of the extreme value problem in the use of Rosenblueth's two-point estimation.

Global path planning is based on an algorithm of optimal path selection which is divided into intelligent and graph search algorithms. Graph search algorithms [10] include Dijkstra's algorithm and A* algorithm. As an A* algorithm uses heuristic estimation, it reduces the amount of search, improves efficiency, and guarantees the optimality of the path. However, the efficiency is lower in a complex environment of a large scale. Intelligent algorithm [11, 12] simulates the biological evolution and biomimetic nature of insect foraging and nesting, mainly including genetic algorithm, ant colony algorithm, particle swarm optimization, etc. This is appropriate for solving and optimizing complex problems by having the characteristics of potential parallelism but has slow operation speed and premature solution and does not consider randomness and time dependence.

Previous studies proposed good solutions for the shortest path problem with randomness but not for the problems of time dependence. A problem of randomness with time dependence requires a different approach to obtain the solution. Therefore, this study aims to find a solution to the trip time and path selection by considering the random processes that change with time, that is, time-dependent. Thus, the shortest path problem is considered as a segmented decision-making problem by using dynamic programming.

3. Problem Descriptions

The real-time navigation of a vehicle requires planning an optimal path from the current to the target position (node) based on real-time information. By updating the trip times in each segment, the guidance strategy shows the shortest path with an appropriate algorithm. When the strategy does not consider the variability of a driving route, it only proposes a suboptimal path. Variability is solved with dynamic programming that decomposes a complex problem into a series of simple problems. The optimal path of a route is obtained by decomposed into segments between the nearest two nodes. For the shortest path problem of the nodes, the principle of stochastic dynamic programming is considered for dynamic path selection.

3.1. Definition of STDSP. The real-time navigation of a vehicle requires planning an optimal path from the current to the target position (node) based on real-time information. By updating the trip times in each segment, the guidance strategy shows the shortest path with an appropriate algorithm. When the strategy does not consider the variability of a driving route, it only proposes a suboptimal path. Variability is solved with dynamic programming that decomposes a complex problem into a series of simple problems. The optimal path of a route is obtained by decomposed into segments between the nearest two nodes. For the shortest path problem of the nodes, the principle of stochastic dynamic programming is considered for dynamic path selection.

The traffic network is a spatiotemporal network. The spatiotemporal network is expressed as

$$G_T = (V_T, A_T), \quad (1)$$

where $V_T = \{(i, t) | i \in V, t \in T\}$ and $A_T = \{(i, t), (j, t + \tau_{ij}(t)) | i, j \in V, t, t + \tau_{ij}(t) \in T\}$.

Here, (i, t) refers to a point of (space, time). The defined G_T does not have a closed-loop and avoids a cyclic search. When the successor node is defined as $\Gamma(i, t)$, the next node is $\Gamma^{-1}(i, t)$. $X_a(t)$ represents the trip time in a segment at time t , and $\mu_a(t)$, $\sigma_a(t)$, respectively, represent the expectation and variance at time t . $f_{X_a(t)}(x)$ represents a probability density function.

There are three possible directions at the intersections: going straight, turning left, and turning right. The signal control system develops from the point control of "single point fixed cycle" to the direction of line control and surface control. The signal cycle of intersection will be adjusted according to the traffic flow. Due to its complexity, most intersections are still controlled by point control in reality, so this article only studies the situation that the signal cycle does not change.

The problem is represented by the average of past historical data as $d_{ij}(t)$ which means the delay of intersection with the next node j on node i at time t . Then, the strategy of the path selection problem is defined as

$$\begin{aligned} p: V_T &\longrightarrow V, \\ p(i, t) &\in \Gamma(i, t). \end{aligned} \quad (2)$$

It is a mapping strategy from the node of a spatio-temporal network to the node of a nontemporal network. When n_k is the k -th node to arrive, and its corresponding arrival time is t_k , then $(n_{(k+1)}, t_{(k+1)}) = p(n_k, t_k)$. $((n_k, t_k), (n_{(k+1)}, t_{(k+1)}))$ is random due to the randomness of the trip time of $t_{(k+1)}$ to node $n_{(k+1)}$. It can be used as a predicted value of the time to reach $n_{(k+1)}$ and from node n_k .

3.2. The Complexity Caused by the Branch of Probability. Since the trip times on a route are random, the time to reach a certain node is uncertain. Thus, total trip time is decomposed into several arrival times with the probability of different shortest paths.

Figure 1 shows an example. If the departure time from node 1 is 0, the probability to reach node 2 at $t = 10$ and $t = 45$ is 0.5, respectively. There are five routes from node 1 to node 6 to choose from. The probability to reach node 6 at $t = 25$ and $t = 60$ is 0.5, respectively, and the expected trip time is 42.5 s.

Path (1, 2, 3, 6) is the same as path (1, 2, 4, 6) but has a different probability and expected trip time of 57.5. The expected trip times of path (1, 3, 6), path (1, 3, 2, 4, 6), and path (1, 5, 6) are 95, 75, and 70, respectively. The paths are not the optimal trip from node 1 to node 6. The trip from node 1 at $t = 0$ to node 2 at $t = 10$, node 3 at $t = 15$, and node 6 at $t = 20$ sequentially has the probability of 0.5. For some reason, when the trip reaches node 2 at $t = 45$ due to the entry time limit on the road section (3, 6) and changes its path to node 4 at $t = 50$ and then to node 6 at $t = 60$, the probability is also 0.5 and the expected trip time from node 1 to node 6 is 40. This path has the shortest time, but it is not the same as the path in the static state that is expressed as a path (1, 2, 3, 6|4, 6).

In this case, the search process of the shortest path from node 1 to node 6 can be resolved by using an AND/OR tree. The decomposition needs to define the state representation of the problem. The problem has two types of states. One is the state when it reaches a certain node, and the other is the selection state when starting from a certain node. The two types of states alternately make a pair. The decomposition process of the original problem is shown in Figure 2. The numbers in gray and white ellipses represent (node, arrival time) and (node, arrival time, next node). The search process starting from node 1 at time 0 includes the subprocesses to nodes 2, 3, and 5, which requires "OR" node. Starting from node 1 to node 2 at time 0, due to the randomness, there are two arrival times to node 2. When there is the shortest path from node 2 to node 6, it is indicated with a certain probability, and the shortest trip time from node 1 to node 6 can be calculated. In this case, this process needs an AND node, and the trip time is the sum of that between the previous nodes. Through AND/OR tree decomposition, every feasible path from node 1 to node 6 can be found, and the optimal path by dynamic path selection at node 2 can also be found by a thick line in Figure 2.

In decomposing the STD shortest path, the AND nodes are not appearing in the static shortest path problem. If dynamic path selection is not performed at node 2, the

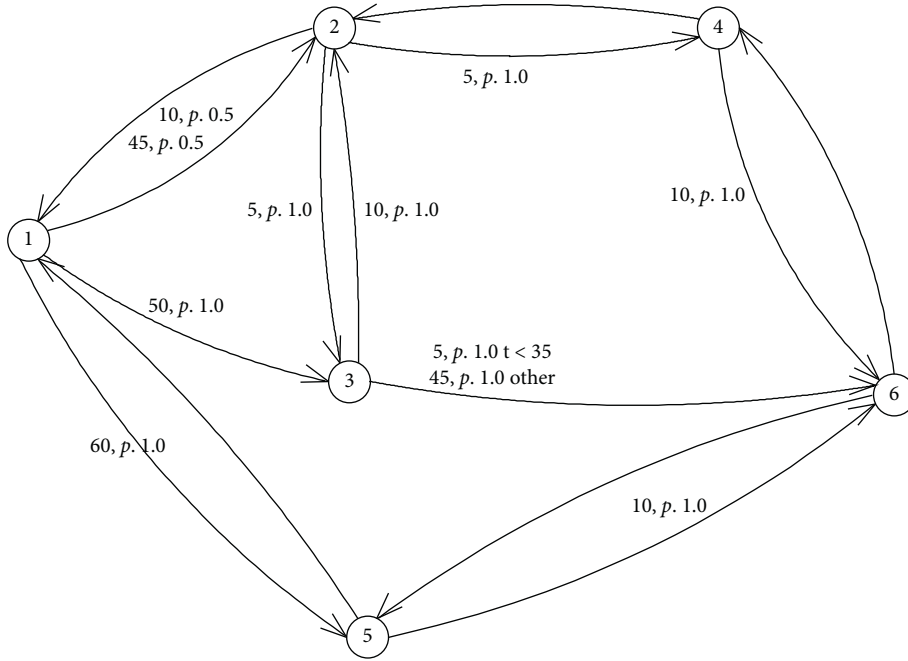


FIGURE 1: An STD net.

optimal route is (1,2,4,6), and the estimated trip time is 42.5 that is longer than that of the optimal path (1,2,3,6|4, 6) with the trip time of 40. This shows that the STD problem is complicated due to time correlation although the estimated trip time is used as the optimal evaluation criterion.

3.3. FIFO Complexity Caused by Unsatisfactory Conditions. Kaufman and Smith [16] defined the FIFO condition when solving the time-dependent shortest path, that is, for any $(i, j) \in A$, $s + d_{ij}(s) \leq t + d_{ij}(t)$ for all $s, t \in T$, when $s \leq t$ is always true. As long as the FIFO conditions are met, efficient labeling algorithms solve the shortest time-dependent path problem.

Since the time-dependent trip time of the (2,3) section does not meet the FIFO condition ($2 + 4$ is not less than $3 + 2$), the shortest path trip time is 6 if the Dijkstra algorithm is used (Figure 3). However, the trip from nodes 1 to 3 has the shortest trip time of 5, which means that nodes 1 to 2 are not the shortest path. This does not conform to the Bellman optimality principle. When the FIFO condition is met (Figure 3(a)), the Bellman optimality principle is also satisfied. Miller-Hooks and Mahmassani [17] extended the FIFO condition with the time-dependency and defined it as

$$\begin{aligned} \forall (i, j) \in A, \\ \text{Prob}\{s + \tau_{ij}(s) \leq t + \tau_{ij}(t)\} = 1, \quad \forall s \leq t, \end{aligned} \quad (3)$$

where $\tau_{ij}(t)$ is the trip time on (i, j) . This implies that the STD network with an independent trip time is calculated with the assumption of consistency between the paths.

In Figure 4(b), since the trip time of (2,3) section does not meet the extended FIFO condition, the expected shortest path from nodes 1 to 3 at time 0 is 5, and its route is $((1,0), (2,3) \&\& (2,4), (3,5))$; the expected shortest path from nodes 1 to 2 at time 0 is 2.5, and its route is $((1,0), (2, 2) \&\& (2,3))$. The shortest path from nodes 1 to 2 is inconsistent with that from nodes 1 to 3, which means that the expected trip time is not simply equal to the expected trip time. The situation that does not meet the definition of the extended FIFO is common in STD networks. Thus, it is unreasonable that the non-FIFO section of the shortest path is excluded.

4. Dynamic Programming Model and AO* Algorithm

4.1. Dynamic Programming Model. An optimal strategy $p^*(i, t) \in \Gamma(i, t)$ of the path from a node to the target node in the shortest trip time is described as follows:

$$\begin{aligned} p^*(i, t) \in \arg \min \{ \mu_a(t) + v^*(j, t + d_{ij}(t) + X_a(t)) + d_{ij}(t) | \forall a \\ = ((i, t), (j, t + d_{ij}(t) + X_a(t))) \text{ and } (j, t + d_{ij}(t) + X_a(t)) \in \Gamma(i, t) \}, \end{aligned} \quad (4)$$

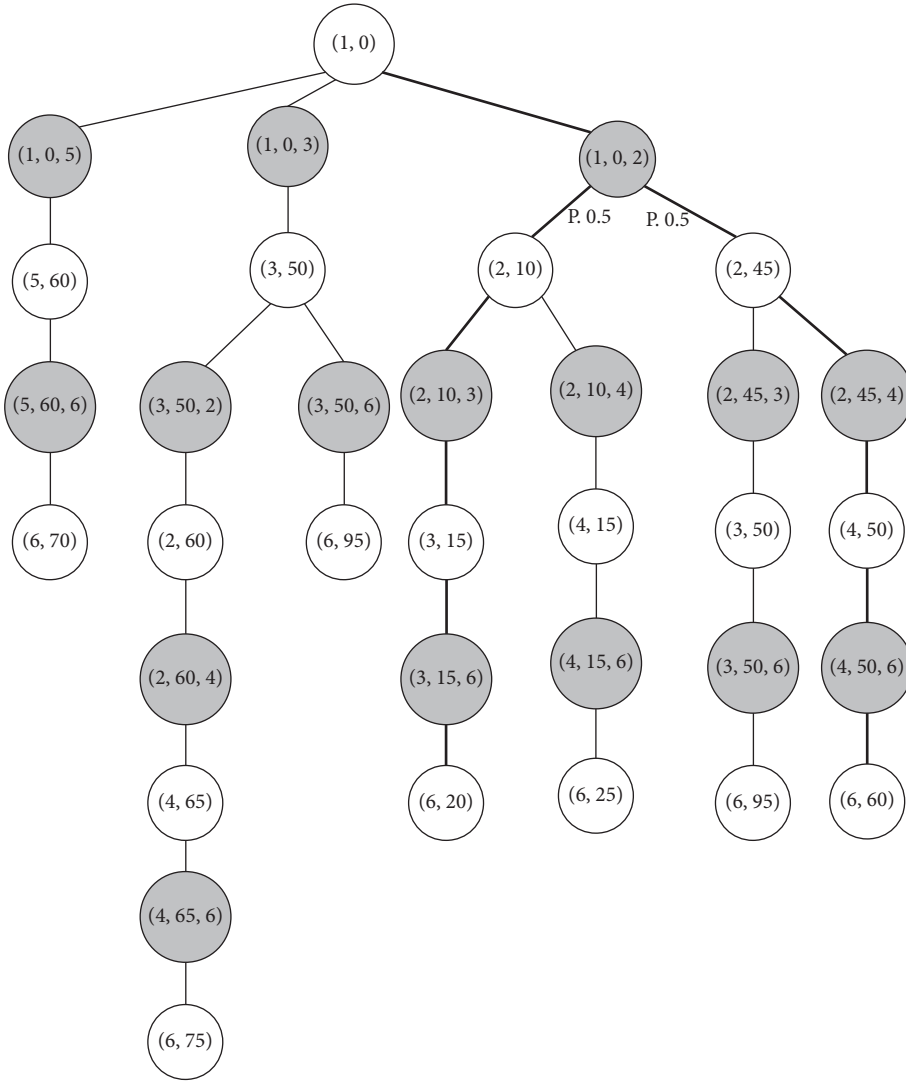


FIGURE 2: STD AND/OR graph decomposition of the shortest path problem.

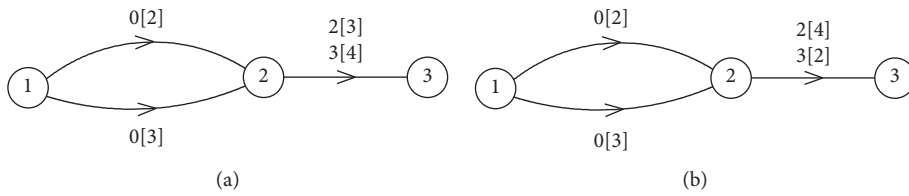


FIGURE 3: Time-dependent dynamic network. (a) FIFO definition. (b) Non-FIFO definition.

where $v^*(i, t)$ is the shortest trip time t from nodes i to d at time t . Then,

$$v^*(i, t) = E(Y_i(t)), \tag{5}$$

$$Y_i(t) = \min \left\{ \begin{array}{l} v^*(j, t + d_{ij}(t) + X_a(t)) + X_a(t) + d_{ij}(t) \\ \forall a = ((i, t), (j, t + d_{ij}(t) + X_a(t))) \text{ and } (j, t + d_{ij}(t) + X_a(t)) \in \Gamma(i, t) \end{array} \right\}, \tag{6}$$

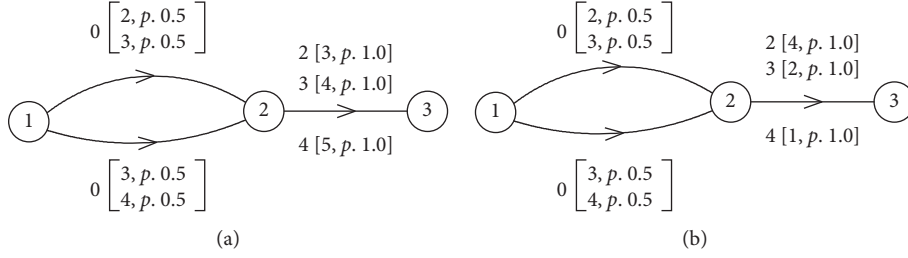


FIGURE 4: Simple random time-dependent net. (a) Meet the definition of extended FIFO. (b) Non-FIFO definition.

with boundary conditions $v^*(d, \cdot) = 0$, the recursive process of dynamic programming is used to solve equations (5) and (6). However, obtaining the random variable is difficult as the extreme function. $Y_i(t)$ has at least one random variable. The statistics $F(y)$ is defined as

$$F_{Y_i(t)}(y) = 1 - [1 - F_{X_{a1}(t)}(y)] [1 - F_{X_{a2}(t)}(y)] \cdots \cdot [1 - F_{X_{a|\Gamma(i,t)|}(t)}(y)], \quad (7)$$

where $|\Gamma(i, t)|$ represents the out-degree of the spatiotemporal node (i, t) . Although the trip time in each independent segment follows an asymptotically normal distribution [25], the random variables do not meet the requirements of independent and identical distribution:

$$E(Y_i(t)) = \int_{-\infty}^{+\infty} y dF_{Y_i(t)}(y). \quad (8)$$

By equation (8), it is very difficult to solve $E(Y_i(t))$. Fu [24], based on Rosenblueth's two-point estimation method, calculates the expectation and variance of random variable function from the expectation and variance of random variable, so as to effectively avoid solving the distribution of random variable function. However, the correctness of the equation is questionable as the extreme values were not differentiable, and Rosenblueth used a series to expand and approximate the first few terms for approximation. As we do not assume the dependence of road trip time on time, the time of entering a segment does not have to be considered for recursing. This study starts from the Rosenblueth two-point estimation method [26] to study the discrete method of solving $E(Y_i(t))$.

4.2. Numerical Stochastic Dynamic Programming Model. The density of the random variable X is defined as the function $f_X(x)$ and the random variable $Z = h(X)$, a function of the random variable. The two-point estimation method uses the probability of two points to express $f_X(x)$, and the probability of these two points meets the first three moments of $f_X(x)$. $E(Z^k)$, $k = 1, 2$, is calculated from two discrete values if $Z = h(X_1, X_2, \dots, X_n)$ is a function of n random variables. Then, $E(Z^k)$, $k = 1, 2$, is calculated from 2^n discrete values.

Figure 5 depicts the typical situation of a route selection when the vehicle is at node (i, t) . At the node, there are three possibilities to move to the next node: going straight, turning left, and turning right.

When $Y_i^k(t)$ is defined as the loop variable with the minimum value of $Y_i(t)$ and the number of loops of k , equation (6) is written as follows:

$$Y_i^1(t) = v^*(j_1, t + \tau_{ij_1}) + X_{ij_1}(t + d_{ij_1}(t)) + d_{ij_1}(t), \quad (9)$$

$$Y_i^k(t) = \min\{Y_i^{k-1}(t), v^*(j_k, t + \tau_{ij_k}) + X_{ij_k}(t + d_{ij_k}(t)) + d_{ij_k}(t)\}, \quad k = 2, \dots, |\Gamma(i, t)|. \quad (10)$$

The expected calculation corresponding to equations (9) and (10) is then

$$\begin{aligned} v_k^*(i, t) &= E[Y_i^k(t)] \\ &= E[\min\{Y_i^{k-1}(t), v^*(j_k, t + \tau_{ij_k}) + X_{ij_k}(t + d_{ij_k}(t)) + d_{ij_k}(t)\}] \\ &= \min\{v_{k-1}^*(i, t), v^*(j_k, t + \tau_{ij_k}) + E[X_{ij_k}(t + d_{ij_k}(t))] + d_{ij_k}(t)\}, \quad k = 2, \dots, |\Gamma(i, t)|, \end{aligned} \quad (11)$$

$$v_1^*(i, t) = E[Y_i^1(t)] = v^*(j_1, t + \tau_{ij_1}) + E[X_{ij_1}(t + d_{ij_1}(t))] + d_{ij_1}(t). \quad (12)$$

The trip time of the shortest path from nodes i to d at time t is calculated as

$$v^*(i, t) = E[Y_i(t)] = v_{|\Gamma(i,t)|}^*(i, t). \quad (13)$$

Equations (10) and (11) describe the reverse recursive process of dynamic programming. In the dynamic path selection, the trip time in a segment $X_a(t)$ is calculated in a continuous random process. The time t changes

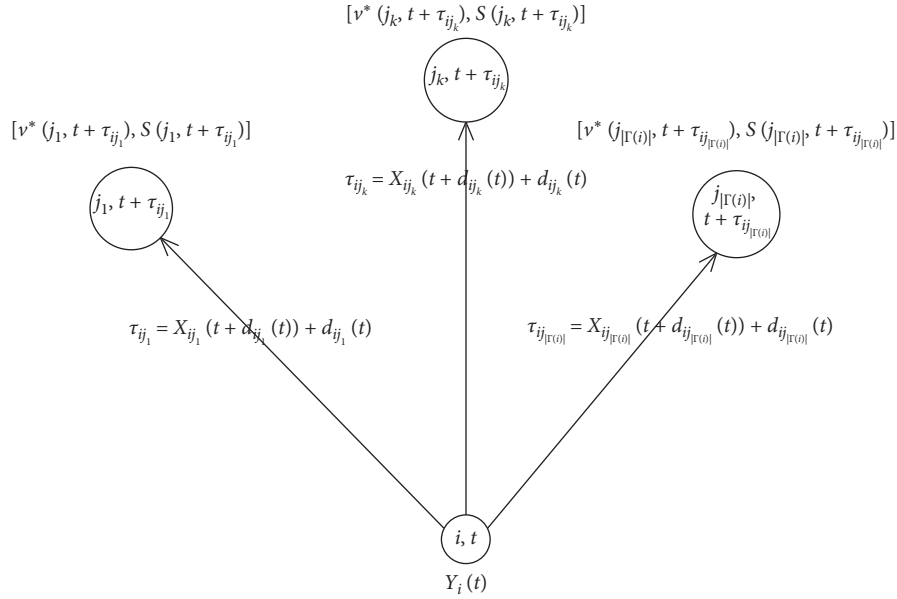


FIGURE 5: Dynamic path choice at node (i, t) .

continuously, and thus the unrestricted boundary condition $v^*(d, \cdot) = 0$ to determine that t has the arbitrariness of the time during the recursion. However, this reverse recursion does not meet the requirement in this study. Thus, the sequential recursive process is adopted.

However, if the trip time in a segment is calculated by a heuristic function instead of $v^*(j_k, t + \tau_{ij_k})$, a sequential recursive process for the path can be obtained from equation (11). If the appropriate heuristic function is used, the optimal path for the search can be determined. Then, when the boundary conditions are reached, $v^*(j_k, t + \tau_{ij_k})$ is deduced along the path, and then $v^*(i, t)$ is deduced inversely.

Rosenblueth's two-point estimation method discretizes the continuous distribution of random variable Z into a simple two-point distribution and is expressed as

$$\begin{aligned} p_1 &= P(Z = \mu_Z - \sigma_Z) = \frac{1}{2}, \\ p_2 &= P(Z = \mu_Z + \sigma_Z) = \frac{1}{2}, \end{aligned} \quad (14)$$

where μ_Z is the expectation and σ_Z is the standard deviation.

The trip time $X_a(t)$ of a segment is discretized into a simple two-point distribution using Rosenblueth's two-point estimation method. That is,

$$\begin{aligned} p_1 &= P(X_a(t) = \mu_{X_a}(t) + \sigma_{X_a}(t)) = \frac{1}{2}, \\ p_2 &= P(X_a(t) = \mu_{X_a}(t) - \sigma_{X_a}(t)) = \frac{1}{2}, \end{aligned} \quad (15)$$

where $h(j, t)$ represents the heuristic trip time in the segment of (j, t) with the probability of 0.5. See

$$c(i, t, j) = \mu_{X_{ij}}(t + d_{ij}(t)) + d_{ij}(t). \quad (16)$$

Equation (11) is discretized as

$$v_k^*(i, t) = \min \left\{ v_{(k-1)}^*(i, t), c(i, t, j_k) + \frac{1}{2} \sum_{m=1,2} h(j_k, t + c(i, t, j_k) + (-1)^m \cdot \sigma_{X_{ij_k}}(t + d_{ij_k}(t))) \right\}, \quad k = 2, \dots, |\Gamma(i, t)|. \quad (17)$$

Similarly, equation (11) is discretized as

$$\begin{aligned} v_1^*(i, t) &= E[Y_i^1(t)] = c(i, t, j_1) + \frac{1}{2} \sum_{m=1,2} h \\ &\quad \left(j_1, t + c(i, t, j_1) + (-1)^m \cdot \sigma_{X_{ij_1}}(t + d_{ij_1}(t)) \right). \end{aligned} \quad (18)$$

Equations (17) and (18) with boundary condition $v^*(d, \cdot) = 0$ constitute a numerical model of stochastic dynamic programming. The solutions of equations (10) and (11) need the selection of heuristic trip times that are modified when the boundary conditions are reached. Dynamic programming is not an algorithm [27], but a modeling method. Based on the heuristic trip time, the AO*

algorithm is effective for solving the random dynamic programming models.

4.3. AO* Algorithm. With the complexity of the STD shortest path problem, an AND/OR tree is used to decompose its solution process. The AND/OR tree inversely calculates the trip time from a precursor node to a successor node in a dynamic programming model that uses equations (17) and (18). The AO* algorithm accompanies an AND/OR graph search in artificial intelligence. Bander and White [28] first applied the algorithm to a nonstatic random shortest path problem and constructed a solution graph. The result was appropriate for a dynamic selection model, as the path was not a single one but a hyperpath. Different path options are decided by the time of entering into a segment.

4.3.1. Graph Heuristic Search and Agreement with AND/OR Graph. The heuristic search algorithm for AND/OR graphs was formally named AO* by Nilsson in 1980 [28]. Whether a node on the AND/OR graph is solved is determined by its successor nodes. An “AND” node becomes solvable only when all its successor nodes are solvable. An “OR” node is solvable when one of the successor nodes is solvable. In the growth process of the graph as the search process, nodes are continuously added to the AND/OR graph. The generated graph is denoted as G_T' and then $G_T' \subseteq G_T$. The demarking process is a retrospective recursive process in which the precursor nodes are solvable by the solvable successor nodes. These are used repeatedly in the search process of the AND/OR graph until a node is marked as a solvable or unsolvable node.

The result of the heuristic search of the AND/OR graph finds the optimal solution graph with the least cost (shortest time). For the AND/OR graph, the nodes that are expected to be part of the optimal solution graph for expansion must be selected. The AND/OR graph with the solvable successor and their precursor nodes is called the hope graph (potential solution graph). In the search process, as new nodes join, the trip time is constantly changing. Therefore, the graph is also constantly changing with the constant graph of heuristic search.

The total cost (time) of the heuristic search of the AND/OR graph is obtained by calculating the trip time in the graph. With $c(i, t, j)$, the trip time from a node (i, t) to a successor node (j, t_j) , the trip time is calculated as follows:

- (1) If the node (i, t) is the target (successor) node, its trip time is a function $f(i, t) = 0$.
- (2) If the node (i, t) is an AND node, its trip time is a function as follows:

$$f(i, t) = \sum_{k=1}^{|\Gamma(i, t)|} cc(i, t, j_k) + f(j_k, t_{j_k}), (j_k, t_{j_k}) \in \Gamma(i, t). \quad (19)$$

- (3) If the node (i, t) is an OR node, its trip time is a function as follows:

$$f(i, t) = \min\{c(i, t, j_k) + f(j_k, t_{j_k}), (j_k, t_{j_k}) \in \Gamma(i, t)\}. \quad (20)$$

- (4) If (i, t) is not scalable and it is not the target node; then, its trip time is defined as $f(i, t) = \infty$.

The trip time of the precursor node is deduced from that of the successor node. Thus, with the optimal solution graph, the trip times from successor nodes to the precursor node are obtained, which is repeated layer by layer, and finally, the starting node can be found. This process solves equations (17) and (18).

4.3.2. Algorithm. With the random variables of the trip time in continuous segments by using the Rosenblueth two-point estimation method, the solution of the random dynamic programming model is regarded as the search process of the AND/OR graph. The flowchart of the AO* algorithm is shown in Figure 6. It includes two processes: generating the successor nodes and the reverse correction of the trip time.

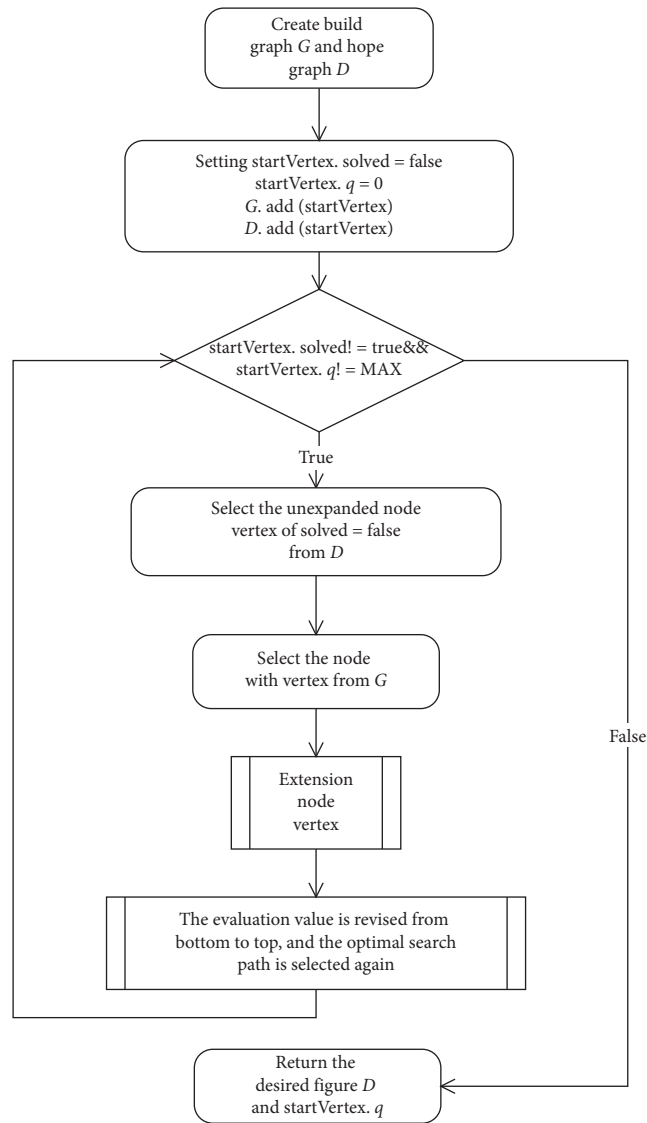
In the algorithm, $q(i, t)$ represents the trip time from a node (i, t) . $h(i, t)$ represents the heuristic function of the node (i, t) . We use the Euclidean distance from a node (i, t) to the target successor node and the maximum free flow velocity in each segment. When the flow speed is zero at the starting node (i, t) ,

$$h(i, t) = \frac{\text{Dist}(i, d)}{u_0}. \quad (21)$$

Meuleau et al. [29] proved that the AO* algorithm must be able to terminate on the optimal solution when the heuristic function calculates the shorter trip time than the real trip time and there is an optimal solution. As the heuristic function in this study satisfies this condition, the optimal decision $p^*(i, t) \in \Gamma(i, t)$ at the node (i, t) is found by the AO* algorithm, which presents the path of the shortest trip time from nodes s to d .

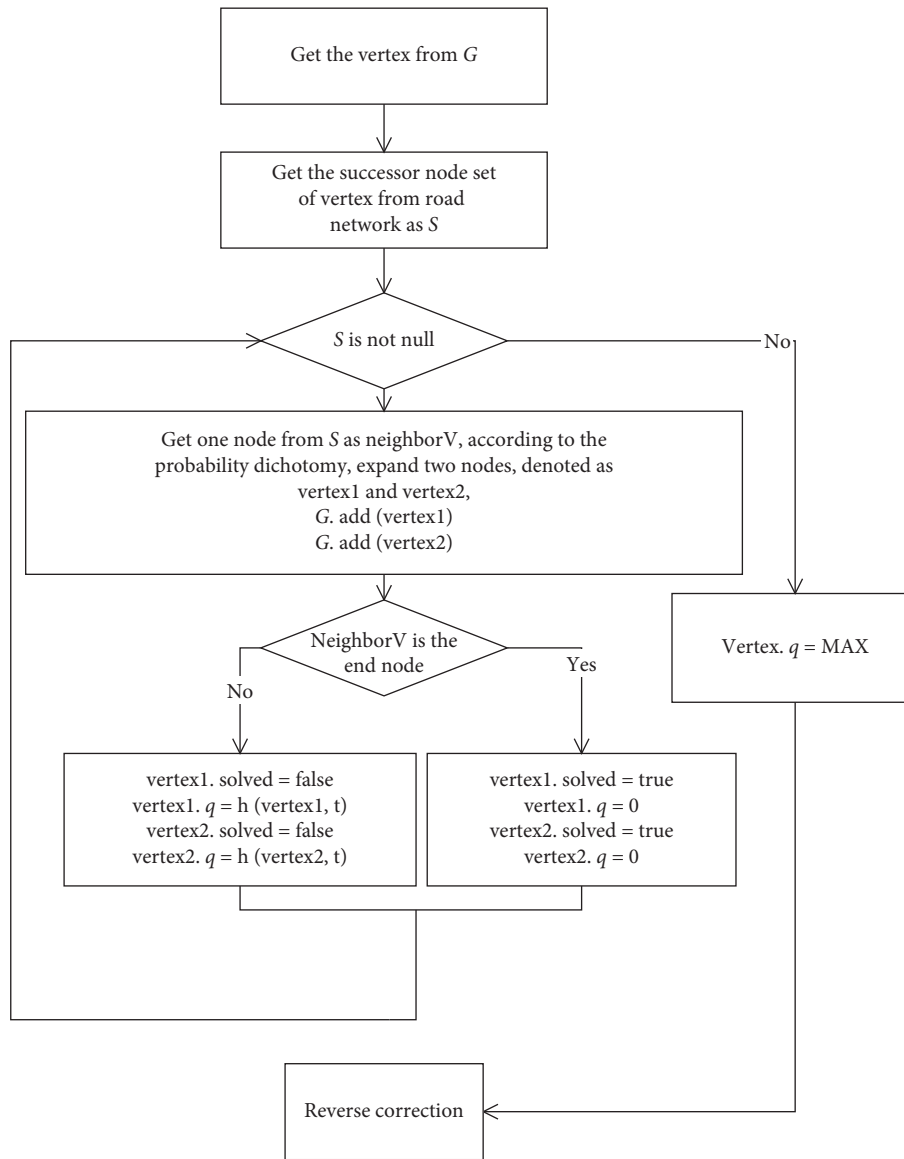
The AO* algorithm does not allow loops in the search process. Finding the following successor nodes requires operating on the nodes that are not the precursors. Due to the monotonicity of time, there is no possibility of loops in the spatiotemporal traffic network G_T . The heuristic search does not find all nodes on G_T but those in the optimal path. The search is processed based upon heuristic information, so the algorithm is efficient in finding the optimal path.

4.4. Improving AO* Algorithm. Since the increased number of successor nodes affects the efficiency of the algorithm, the probability tree diagram is necessary to consider during the search process. When the variance of the trip times in segments is not large, the conditional probability has little effect on the final optimal path selection. In some segments, the past trip times are almost constant. In this case, the conditional probability does not affect significantly. When the variance becomes large, the conditional probability of the branches is considered, which simplifies the process significantly with a less number of nodes and the improvement of the efficiency of the algorithm. Figure 7 shows



(a)

FIGURE 6: Continued.



(b)

FIGURE 6: Continued.

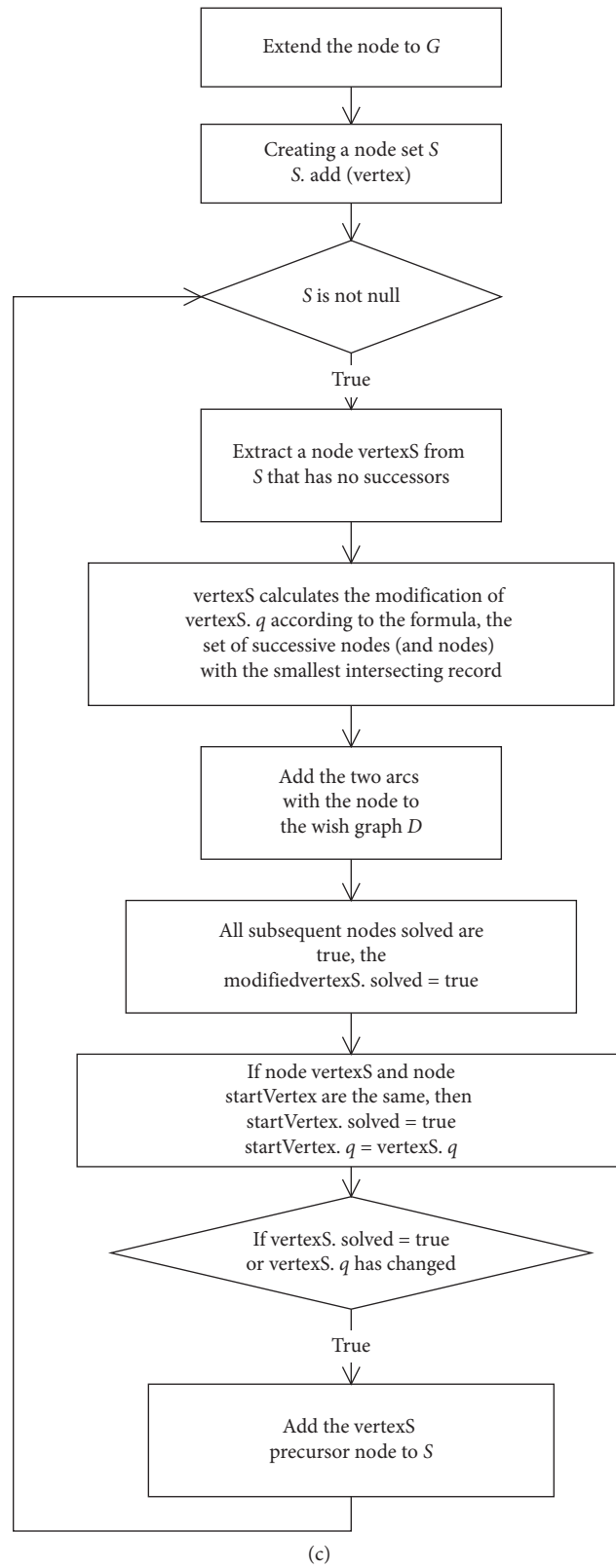


FIGURE 6: Heuristic AND/OR graph search flowchart. (a) General flowchart. (b) Node expansion process. (c) Reverse correction process.

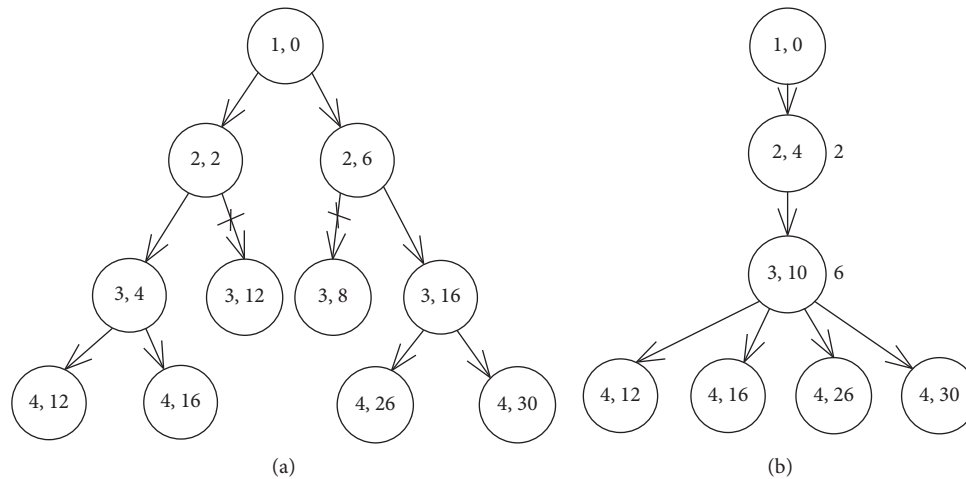


FIGURE 7: Judging the node expansion diagram based on the variance and trip time. (a) Multilevel expansion of the nodes. (b) Limiting the expansion.

diagrams on how to judge for expanding the nodes (increasing the number of nodes) based on the variance of trip times. The branch in Figure 7(a) has the multilevel expansion of the nodes, while that in Figure 7(b) limits the expansion. In each node (circle), the trip times and the variance of them are shown. Figure 7(b) shows that limiting the expansion only needs the nodes with either the longest or the shortest trip time with the estimated trip time unchanged when compared to Figure 7(a) that has the expansion of the nodes. Based on this result, the flowchart in Figure 6 is modified and shown in Figure 8.

5. Case Studies and Discussion

5.1. Application of Improved AO* Algorithm. The traffic data in Shenzhen was used to calculate the trip times of each segment on a route with the improved AO* algorithm. Figure 7 shows the expected trip times (in parenthesis) on various routes and times in a day. In nonpeak times such as 7:00, 9:30, 14:00, and 20:30, the path was chosen for a smaller number of segments than in peak times such as 8:30, 11:00, 12:30, and 18:00 when the road has traffic congestion. When there are many vehicles in a segment, another segment with fewer vehicles is chosen for a shorter trip time.

The calculation result at 12:30 (Figure 9(e)) suggested two different choices at node 138. Of course, all the nodes shown in Figure 9 have the same kind of selection as node 138, but the result of selection is the same road segment in space, and the difference is only in time. It can be considered that the difference in time is only the difference in “quantity,” and the accumulation of “quantity” will cause the change in “quality,” such as the appearance of different road segment selection results. If there is no choice for a

segment, then this means that the change in “quantity” is not enough to cause that in “quality.” If there are two paths on which it takes 45 (going straight) and 50 seconds (turning right) to reach node 138 from node 28, the optimal path will be the path for a shorter total trip time. Of course, the path choice at node 138 depends on the road condition of the segment (nodes 28-138) and the estimated arrival time at node 138.

5.2. Performance of Improved AO* Algorithm. Randomness changes the trip time even in the same segment at the same period. Then, many solution graph nodes need to be considered in the search of the algorithm. Table 1 shows the statistics of the AO* algorithm search process. When the number of nodes is increased, the number of solution graph nodes and the time for search increased. That is, the efficiency of the search decreased. The expansion of solution graph nodes affects the efficiency of the algorithm based on the probability of each branch (the branch).

Table 2 shows the result from the search process by using the improved algorithm. The increase of the number of nodes does not affect significantly the numbers of the solution graph nodes, the generated nodes, the iteration, and the time for search. This indicates the advantages of the heuristic search and the function used.

The data relationship in the two tables is shown in Figure 10. It can be clearly seen that the efficiency of the unrestricted algorithm decreases rapidly as the number of road network nodes increases. The efficiency of the improved algorithm does not change much with the increase in the number of road network nodes, which is also the characteristic of heuristic search.

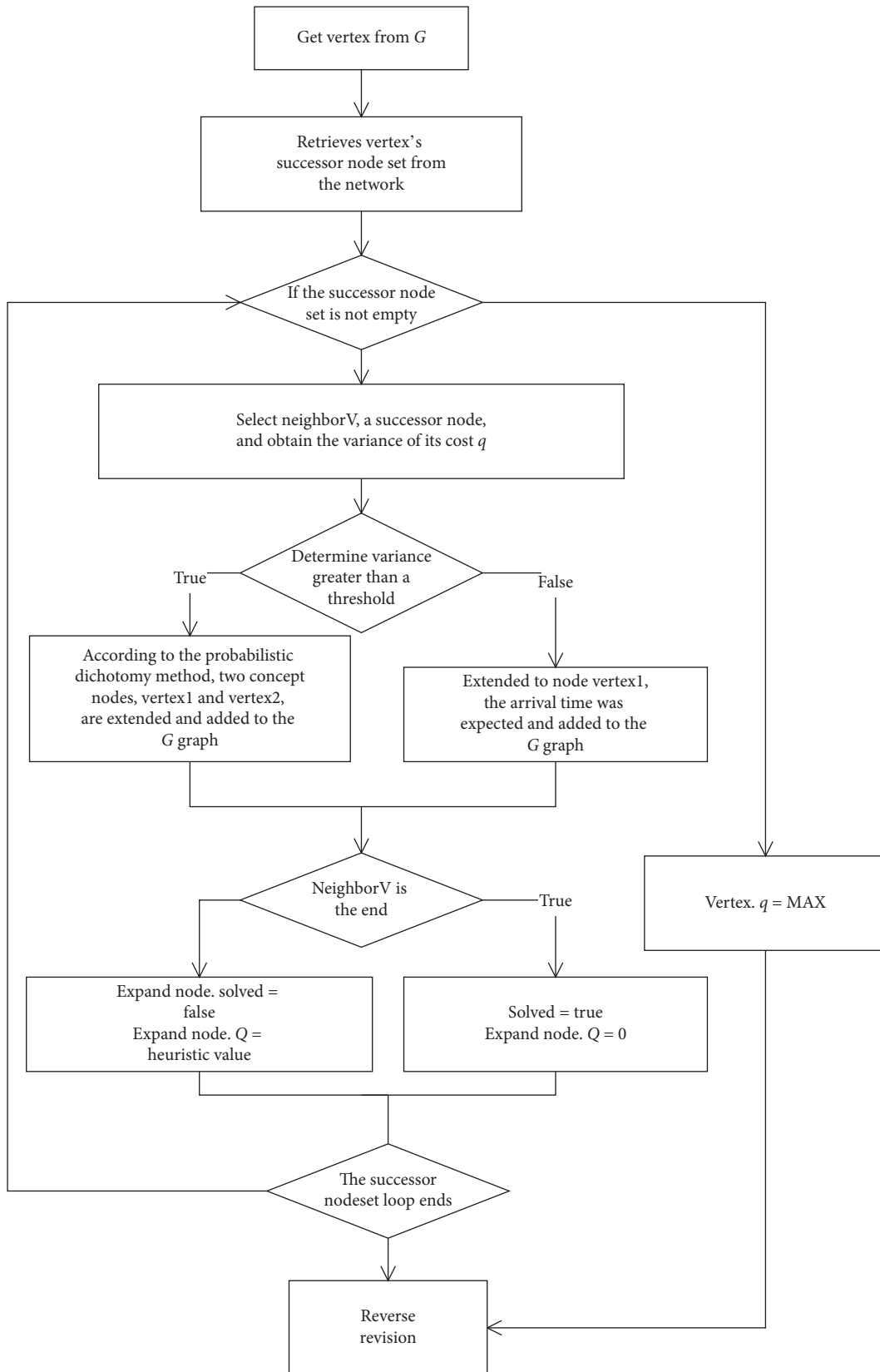


FIGURE 8: The flowchart for the limited expansion search process.

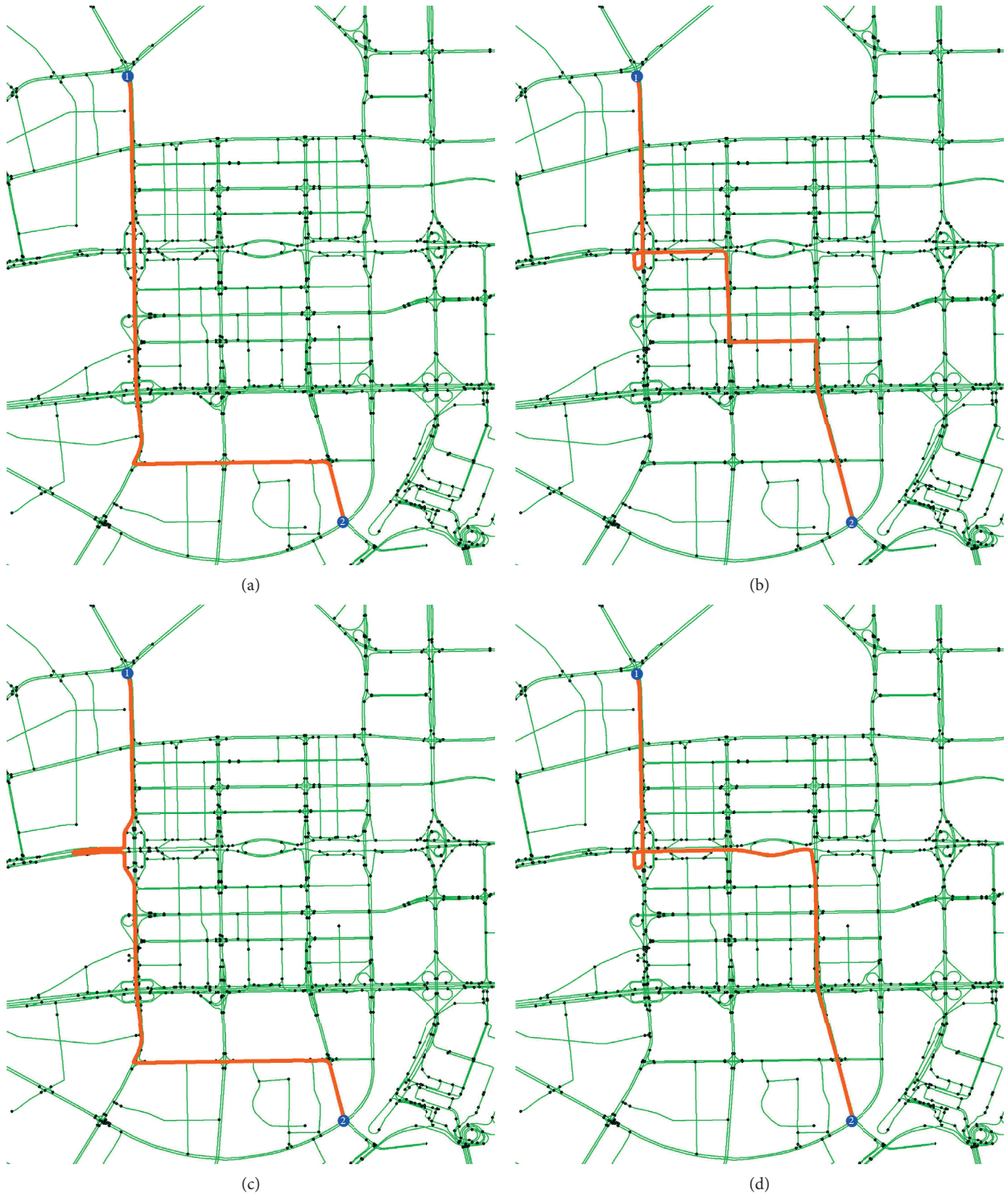


FIGURE 9: Continued.

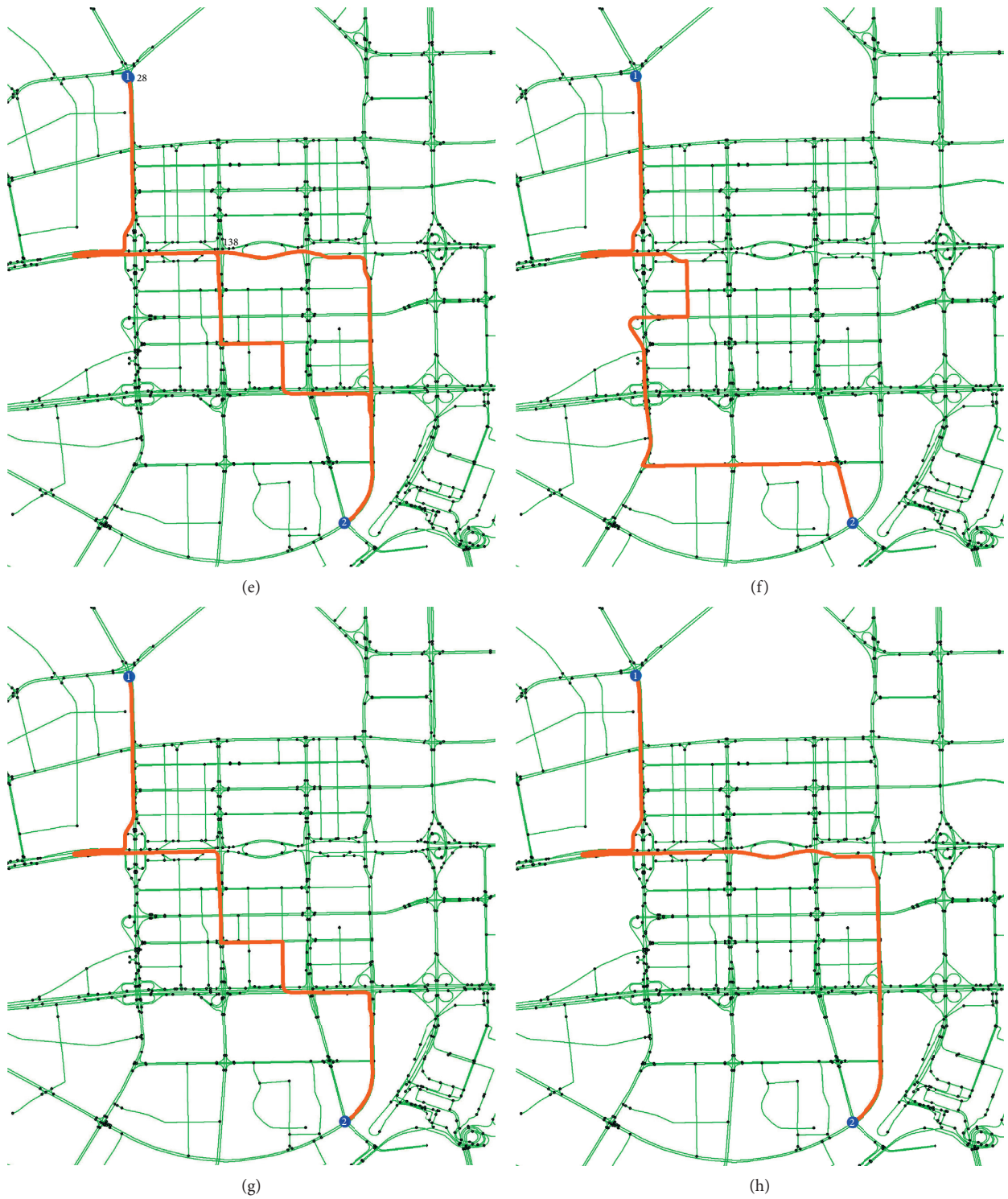


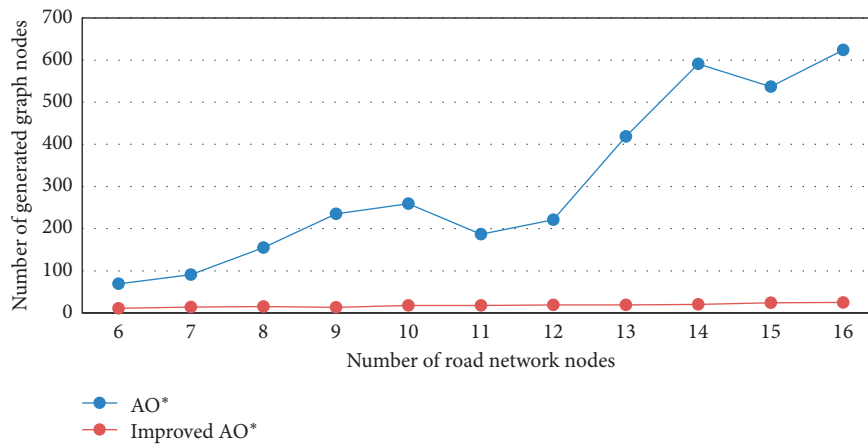
FIGURE 9: Path selection result: (a) at 07:00 (292 s), (b) 08:30 (352 s), (c) 09:30 (320 s), (d) 11:00 (340 s), (e) 12:30 (339 s), (f) 14:00 (318 s), (g) 18:00 (343 s), and (h) 20:30 (300 s).

TABLE 1: The result of the AO* algorithm search.

Number of nodes in an optimal path	Number of solution graph nodes	Number of generated graph nodes	Number of iterations from the starting node	Time for search (s)
6	33	69	17	0.7
7	45	91	26	1.1
8	63	155	29	2
9	97	235	59	4.2
10	100	259	51	4.3
11	96	187	49	4.7
12	78	221	51	8.1
13	145	419	47	17.1
14	226	591	78	34.1
15	220	537	81	49.5
16	253	624	89	51

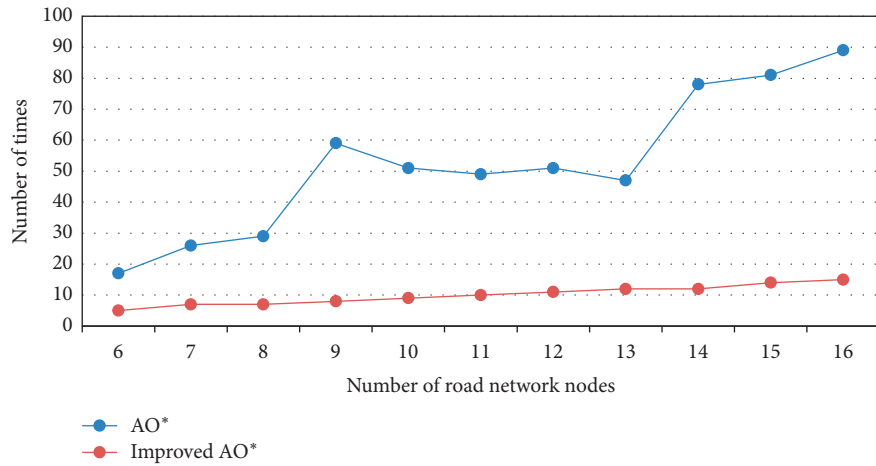
TABLE 2: The result from the improved AO* algorithm search.

Number of nodes in an optimal path	Number of solution graph nodes	Number of generated graph nodes	Number of iterations from the starting node	Time for search (s)
6	7	11	5	0.2
7	8	14	7	0.3
8	8	15	7	0.3
9	9	13	8	0.3
10	10	18	9	0.4
11	11	18	10	0.4
12	12	19	11	0.4
13	15	19	12	0.4
14	14	20	12	0.4
15	17	24	14	0.5
16	16	25	15	0.5

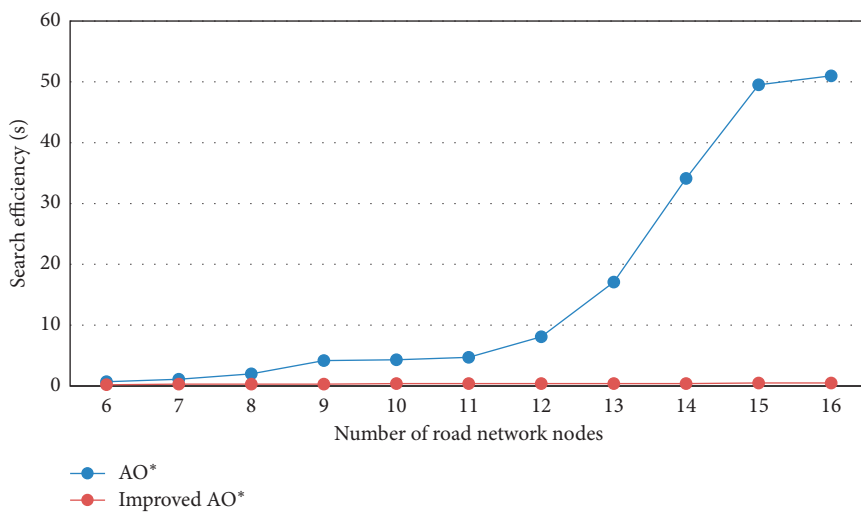


(a)

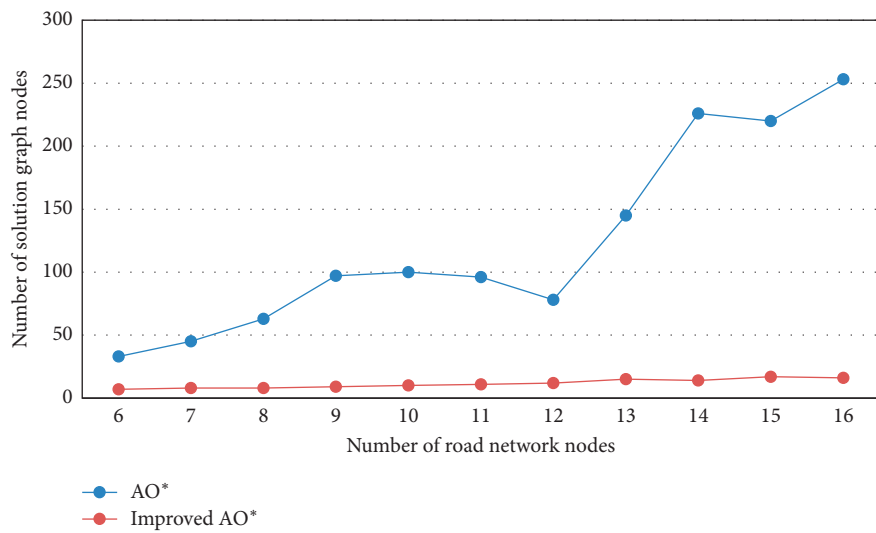
FIGURE 10: Continued.



(b)



(c)



(d)

FIGURE 10: The comparison of the efficiency of the original and improved AO* algorithm. (a) Comparison of the generated graph nodes. (b) Comparison of iteration from the starting node. (c) Comparison of running time. (d) Comparison of solution graph nodes.

6. Conclusion

The trip times in the segments on a route by a vehicle are estimated in a random process with randomness. Each node has a different probability of arriving at the target node in the shortest time according to the entry time. To calculate the trip time, a new algorithm of the optimal path selection was proposed and the following were investigated.

- (1) The complexity of a random dynamic shortest path problem is necessary to consider. The variability of the trip time between nodes is caused by the insufficiency of FIFO to explain the complexity. The dynamic model of the shortest path selection considers the fact that the shortest trip time on a route is not equal to the sum of the trip times in the segments.
- (2) The random shortest path selection is rather a decision-making problem. That is, at each segment, the choice of going straight, turning left, and turning right needs to be made according to the estimated shortest trip time. Stochastic dynamic programming is required to model for the solution of such a stochastic decision-making problem.
- (3) The trip times between nodes on a route are not discrete, and that in the shortest path is the extreme value function with random variables. As the calculation of the probability density of the trip times is complicated, this study adopted Rosenblueth's two-point estimation method. The trip time as a random variable is discretized so that the expectation of the shortest trip time is unchanged.
- (4) The probability density affects the "quantity" of the estimation of the trip times, not the "quality." The "quantity" influences the efficiency of an algorithm. The "quantity" does not mean the increasing number of nodes but possible successor nodes. The increase in the number of selections affects the efficiency of the estimation of the trip time significantly. Therefore, we proposed the improved AO* algorithm for enhancing the efficiency for finding the optimal trip time to apply the algorithm to the real data process. The result proves the superiority of the improved AO* algorithm and the basis for applying this to the real-time navigation system of a vehicle.

Data Availability

The data can be accessed at <https://github.com/ricebow/Improved-AND-OR-Tree-Search>. There are no restrictions on data access. The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This study was supported by the Fujian Province Natural Fund Project (Grant no. 2020J01263), Science and Technology Planning Foreign Cooperation Project of Longyan (Grant no. 2019LYF7003), Open Fund Project of Fujian University Engineering Research Center for Disaster Prevention and Mitigation of Southeast Coastal Engineering Structure of Putian University (Grant no. 2019005), Open Foundation Project of Fujian Provincial Key Laboratory of Higher Education (Putian University) (Grant no. ST19004), and High-Level Talents Program of Xiamen University of Technology (YKJ20010R).

References

- [1] Q. Li and D. Li, "Big data GIS," *Geomatics and Information Science of Wuhan University*, vol. 39, no. 6, pp. 641–644, 2014.
- [2] H. Lu, Z. Sun, and W. Qu, "Big data and its applications in urban intelligent transportation system," in *Proceedings of the 3rd International Conference on Intelligent Transportation, Big Data & Smart City (ICITBS 2018)*, Xiamen, China, January 2018.
- [3] S. Wan, Y. Zhao, T. Wang, Z. Gu, Q. H. Abbasi, and K.-K. R. Choo, "Multi-dimensional data indexing and range query processing via Voronoi diagram for internet of things," *Future Generation Computer Systems*, vol. 91, pp. 382–391, 2019.
- [4] C. Chen, B. Liu, S. Wan, P. Qiao, and Q. Pei, "An edge traffic flow detection scheme based on deep learning in an intelligent transportation system," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 3, pp. 1840–1852, 2021.
- [5] Y. Luo, S. Shao, and Y. Zhang, "Location and path planning of mobile robots based on data fusion," *Journal of Computer Applications*, vol. 30, no. 11, pp. 3091–3093, 2010.
- [6] W. Xu, J. Pan, J. Wei, and J. M. Dolan, "Motion planning under uncertainty for on-road autonomous driving," in *Proceedings of the 2014 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2507–2512, Hong Kong, China, June 2014.
- [7] Y. Rasekhipour, A. Khajepour, S.-K. Chen, and B. Litkouhi, "A potential field-based model predictive path-planning controller for autonomous road vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 5, pp. 1255–1267, May 2017.
- [8] Y. F. Chen, M. Liu, M. Everett, and J. P. How, "Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning," in *Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 285–292, Singapore, May 2017.
- [9] T. Y. Abdalla, A. Abed, and A. A. Ahmed, "Mobile robot navigation using PSO-optimized fuzzy artificial potential field with fuzzy control," *Journal of Intelligent and Fuzzy Systems*, vol. 32, no. 6, pp. 3893–3908, 2017.
- [10] S. M. Persson and I. Sharf, "Sampling-based A* algorithm for robot path-planning," *The International Journal of Robotics Research*, vol. 33, no. 13, pp. 1683–1708, 2014.
- [11] C.-C. Tsai, H.-C. Huang, and C.-K. Chan, "Parallel elite genetic algorithm and its application to global path planning for autonomous robot navigation," *IEEE Transactions on Industrial Electronics*, vol. 58, no. 10, pp. 4813–4821, 2011.
- [12] A. Lissovoi and C. Witt, "Runtime analysis of ant colony optimization on dynamic shortest path problems," *Theoretical Computer Science*, vol. 561, pp. 73–85, 2015.

- [13] Q. Li, H. Li, Z. Xie, and D. Xu, "On the road trip time for dynamic route choice," *Geomatics and Information Science of Wuhan University*, vol. 31, no. 6, pp. 519–522, 2006.
- [14] R. W. Hall, "The fastest path through a network with random time-dependent travel times," *Transportation Science*, vol. 20, no. 3, pp. 182–188, 1986.
- [15] R. Bellman, "On a routing problem," *Quarterly of Applied Mathematics*, vol. 16, no. 1, pp. 87–90, 1958.
- [16] J. Wu, S. Jin, H. Ji, and T. Srikanthan, "Algorithm for time-dependent shortest safe path on transportation networks," *Procedia Computer Science*, vol. 4, no. 4, pp. 958–966, 2011.
- [17] E. D. Miller-Hooks and H. S. Mahmassani, "Least expected time paths in stochastic, time-varying transportation networks," *Transportation Science*, vol. 34, no. 2, pp. 198–215, 2000.
- [18] S. Gao and I. Chabini, "Optimal routing policy problems in stochastic time-dependent networks," *Transportation Research Part B: Methodological*, vol. 40, no. 2, pp. 93–122, 2006.
- [19] X. Wu and Y. Nie, "Modeling heterogeneous risk-taking behavior in route choice: a stochastic dominance approach," *Transportation Research Part A: Policy and Practice*, vol. 45, no. 9, pp. 896–915, 2011.
- [20] A. Khani and S. D. Boyles, "An exact algorithm for the mean-standard deviation shortest path problem," *Transportation Research Part B: Methodological*, vol. 81, no. 4, pp. 252–266, 2015.
- [21] Y. Nie and X. Wu, "Shortest path problem considering on-time arrival probability," *Transportation Research Part B: Methodological*, vol. 43, no. 6, pp. 597–613, 2009.
- [22] M. Steinmetz, J. Hoffmann, and O. Buffet, "Goal probability analysis in probabilistic planning: exploring and enhancing the state of the art," *Journal of Artificial Intelligence Research*, vol. 57, pp. 229–271, 2016.
- [23] L. Fu and L. R. Rilett, "Expected shortest paths in dynamic and stochastic traffic networks," *Transportation Research Part B: Methodological*, vol. 32, no. 7, pp. 499–516, 1998.
- [24] L. Fu, "An adaptive routing algorithm for in-vehicle route guidance systems with real-time information," *Transportation Research Part B: Methodological*, vol. 35, no. 8, pp. 749–765, 2001.
- [25] K. Jeffrey P, "Stochastic analysis of link trip times-distributions, moments, and approximations," Doctoral Dissertation, The Pennsylvania State University, State College, PA, US, 2001.
- [26] Y. Zhao and T. Ono, "New point estimates for probability moments," *Journal of Engineering Mechanics*, vol. 126, no. 4, 2000.
- [27] J. Zietz, "Dynamic programming: an introduction by example," *The Journal of Economic Education*, vol. 38, no. 2, pp. 165–186, 2007.
- [28] J. L. Bander and C. C. White, "A heuristic search approach for a nonstationary stochastic shortest path problem with terminal cost," *Transportation Science*, vol. 36, no. 2, pp. 218–230, 2002.
- [29] N. Meuleau, E. Benazera, R. I. Brafman, E. A. Hansen, and Mausam, "A heuristic search approach to planning with continuous resources in stochastic domains," *Journal of Artificial Intelligence Research*, vol. 34, pp. 27–59, 2009.

Research Article

Learning Behavior Analysis Using Clustering and Evolutionary Error Correcting Output Code Algorithms in Small Private Online Courses

Shu-tong Xie ^{1,2}, Qiong Chen,³ Kun-hong Liu ⁴, Qing-zhao Kong,⁵ and Xiu-juan Cao¹

¹School of Computer Engineering, Jimei University, Xiamen 361021, China

²Digital Fujian Big Data Modeling and Intelligent Computing Institute, Jimei University, Xiamen 361021, China

³College of Computer and Information Sciences, Fujian Agriculture and Forestry University, Fuzhou 350002, China

⁴School of Informatics, Xiamen University, Xiamen 361005, China

⁵School of Science, Jimei University, Xiamen 361021, China

Correspondence should be addressed to Kun-hong Liu; lkhqz@xmu.edu.cn

Received 22 March 2021; Accepted 2 June 2021; Published 14 June 2021

Academic Editor: Zhaoqing Pan

Copyright © 2021 Shu-tong Xie et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

In recent years, online and offline teaching activities have been combined by the Small Private Online Course (SPOC) teaching activities, which can achieve a better teaching result. Therefore, colleges around the world have widely carried out SPOC-based blending teaching. Particularly in this year's epidemic, the online education platform has accumulated lots of education data. In this paper, we collected the student behavior log data during the blending teaching process of the "College Information Technology Fundamentals" course of three colleges to conduct student learning behavior analysis and learning outcome prediction. Firstly, data collection and preprocessing are carried out; cluster analysis is performed by using k-means algorithms. Four typical learning behavior patterns have been obtained from previous research, and these patterns were analyzed in terms of teaching videos, quizzes, and platform visits. Secondly, a multiclass classification framework, which combines a feature selection method based on genetic algorithm (GA) with the error correcting output code (ECOC) method, is designed for training the classification model to achieve the prediction of grade levels of students. The experimental results show that the multiclass classification method proposed in this paper can effectively predict the grade of performance, with an average accuracy rate of over 75%. The research results help to implement personalized teaching for students with different grades and learning patterns.

1. Introduction

In recent years, with the rapid development of Massive Open Online Course (MOOC), millions of students can have a more convenient and efficient learning experience [1]. However, MOOC always serves tens of thousands of students at the same time, which cannot satisfy the individual needs of the specific student group because they have different levels of learning skills. To address the problem, Small Private Online Course (SPOC) was developed as an alternative which can provide the learning resource in MOOC and perform the interaction between instructors and students in offline class [2]. Therefore, it is an important academic and educational problem to find the relationship

between the learning behavior and learning outcome by machining learning methods, which analyze the log data of student learning behaviors on both MOOC platform and offline class [3, 4].

In this paper, we study how students engage in SPOCs through clustering and multiclassification algorithms and the relationship between their learning pattern and learning outcome. The main contributions of this paper are as follows:

- (1) Previous researches mainly focus on log analysis of MOOC. But the SPOC teaching mode is the important teaching method implemented by colleges in recent years because it combines online MOOC with

offline traditional class and can achieve a better learning outcome than traditional MOOC. The study of SPOC-based learning behavior analysis is more conducive to the improvement of teaching effects.

- (2) The clustering algorithm is used to find out the learning behavior patterns of students; combined with years of teaching experience, four typical learning behavior patterns are summarized; the characteristics of different learning patterns are analyzed to provide the basis for individual teaching.
- (3) Previous studies focus on predicting students' dropout, obtaining certificates, and other binary classification problems. This paper studies four classification problems (excellent, good, pass, and fail), which are helpful to distinguish the learning levels of students and lay the foundation for personalized teaching.
- (4) The multiclass classification framework that combines the error correcting output code (ECOC) method and genetic algorithm (GA) obtains good prediction performance.

The rest of the paper is organized as follows. Section 2 describes the related work on education data mining and learning analytics. Section 3 briefly introduces data sources and the preprocessing. In Section 4, four typical learning behaviors are presented in detail. Section 5 presents the proposed multiclassification approach. Section 6 compares the proposed approach with others. Finally, Section 7 draws conclusions and provides the future research trends.

2. Literature Review

There are several successful cases of how MOOC can be combined with flipped classroom [5–7]. A lecturer at the University of Regina tested the idea in one of his Electronic Systems Engineering courses. Students watched videos of lecture at their own time and own pace before the class time. Then they were assigned into groups to complete group and individual assignments in the classroom. The lecturer kept on monitoring the group performance and interacting with his students. The performance of students in this preliminary research was found to be positive. The lecturer summed up his experience that the nature of in-class assignment was critical to success. Assignments relying heavily on group interaction usually lead to an outcome of higher ability students dominating less able students. He suggested that individual assessments with group interactions may have an overall involvement [6].

In another case, Ng et al. studied the student readiness and their learning achievements after adopting flipped classroom teaching in a database class [7]. They analyzed the command types of each clickstream and the relationship between the activity rate in MOOC and midterm test results. The results showed a positive relationship between students' activity rate in MOOC and their academic outcome.

On the other hand, a study conducted by the Stanford and Cornell University showed a positive relationship

between the student engagement and the exam results on MOOC [8]. However, the linear relationships only hold up to a certain point. When the activity level increased around 80% score, the examination results showed nondirect association and the trend of diminishing marginal returns was applied with respect to the examination grade. In 2015, Hughes et al. conducted a research to analyze the attendance on MOOC and the impact on the final grades. The research treated the overall module attendance as the students' engagement and found a generally positive trend of students' attendance and examination outcome [9].

Recently, Qiu et al. analyzed the students' demographics and learning activities including forum activities, time spent on videos, and assignments, which influenced students' engagement in MOOC through the ordinary least squares (OLS) models [10]. A latent dynamic factor graph (LadFG) model was also proposed to predict the students' assignment grade as well as certificate earners. The proposed approach performed better than several previous alternative predictive approaches, that is, logistic regression classifier (LR) [11], support vector machine (SVM), and factorization machines. Similarly, Xu and Yang [12] presented a two-step approach to find the surrogate exam-takers on MOOC, which, in nature, is a certificate earner prediction approach. They developed a function to divide all learners into three groups including certificate earning, video watching, and course sampling. Then an SVM-based algorithm was proposed to predict the certification earners in the certificate earning group. Bailey et al. [13] developed a logistic regression-based approach to predict the probabilities of assignments grade weekly to intervene the at-risk students, who were nearby the pass/fail border. In order to trade-off smoothness and accuracy, two proposed LR-based transfer learning algorithms were added a regularization term to minimize the difference of failure probability between consecutive weeks. Ezen-Can et al. [14] proposed an algorithm that combined a k-medoids clustering with the greedy seed selection method to understand the discussion forums automatically on MOOC, which allows automated discourse analysis and mining to better support students' learning. Zhang et al. collected multisource heterogeneous data from two MOOC courses, "Data Structures and Algorithms" and "Introduction to Computers." Then they analyzed students' learning content by identifying important concepts in the courses, students' knowledge was evaluated through students' test data, and algorithms were adopted to predict whether students dropout. By doing so, personalized learning guidance is available by analyzing multisource data in MOOCs [15]. Yu et al. identified seven cognitive participation models of students based on their video clickstream logs and designed a classification algorithm based on k-nearest neighbor (KNN), SVM, and artificial neural network (ANN) to predict whether students can pass the exam [16].

In order to improve the teaching efficiency of traditional offline courses and MOOC courses, Meier et al. used course historical teaching data, including homework, quizzes, and midterm exams, to predict the performance of students in subsequent learning output (good/poor), to gain time for

intervention in the early stage of teaching for poor students in UCLA [17]. Xu et al. designed an ensemble classification system with a two-layer structure to predict the changing learning status of students dynamically. And they proposed a data-driven method based on the latent factor model and probability matrix decomposition and found the relevance of the curriculum. By doing so, the forecast accuracy was improved [18]. Ulloa-Cazarez et al. [19] proposed a genetic programming (GP) algorithm to predict whether students can pass the final exam. In order to identify students who have difficulty in learning the “Digital Design” course, Hussain et al. developed the ANN and SVM by using the learning behavior data on the learning system to predict students with a learning difficulty, which, in turn, facilitates early teaching intervention [20].

3. The Data Set and the Preprocessing

Three online courses were involved in this study. We try to acquire learning behavior data from these students, such as video watching times, access parameter in MOOC (i.e., frequency and time slot), homework/test score, and the final course grades as well. Our work does not only provide an effective experimental review for the use of SPOC in different schools but also help to find out effective learning patterns of student in common.

The selected course was the “College Information Technology Foundation,” which is a compulsory subject for the undergraduate students in three schools. The course contents included computer hardware and software, database, multimedia, information security, and network. The course adopted the SPOC teaching mode which combines with the MOOC and offline traditional classroom. Students studied the basic knowledge by watching several short videos on the MOOC platform before traditional offline class. Then flipped class was executed. Furthermore, students were required to complete eight assignments and eight tests on the MOOC provided by the Chaoxing Company. There were 2,439 students from school A, 3,119 students from school B, and 3612 students from school C who participated in the course.

In the data set, the data of school A contains 43 fields, school B contains 89 fields, and school C contains 200 fields. The fields include basic information about students and teachers, viewing of teaching videos, completion of students’ tests, and final grades. Firstly, the unrelated features and redundant features are eliminated by feature selection, including information such as student number, professional class number, teacher ID, and discussion number. Secondly, the fields of data set are processed and converted, and then all features are normalized, including the watching time of each video, score of the homework/test, and the submission time of each test. Finally, the samples of students who had not watched the lecture video or completed the test on the MOOC platform are defined as invalid samples. The invalid samples, in turn, are removed during the data preprocessing. After data preprocessing, school A retains 2430 records containing 35 fields, school B retains 3107 records containing 82 fields, and school C retains 3612 records

containing 153 fields. Finally, the data is classified into four classes according to the final grade. Class 4 is the students whose final score is between 0 (including 0) and 60, Class 3 is the students whose final score is between 60 (including 60) and 75, and Class 2 is the students whose final score is between 75 (including 75) and 85. And the students whose final score is between 85 (85) and 100 are classified as Class 1. Table 1 shows the number of students for different classes in three schools.

4. Learning Behavior Analysis by Clustering Algorithms

Video watching time and tests completed are two essential aspects of student learning behavior which can affect the learning performance. Data related to video viewing behaviors includes total video viewing time and the percentage of viewing time of video 1 to video 11. The data related to tests’ completed behaviors include test progress, scores from test 1 to test 8, and submission time from test 1 to test 8. In addition, the number of visits to the student’s MOOC platform also reflected the student’s learning behavior in some way. Clustering algorithm has been succeeding application in bibliometric data analysis, text classification, and so on. In order to identify the learning behavior patterns, the k-means algorithm is adopted to cluster students with different learning behavior patterns from school A; combining the experience of years in blended teaching, four typical learning behavior patterns were found, as shown in Table 2. The similar learning behavior patterns can also be found in schools B and C, but the proportion of the four types of students is different from that of school A due to different data distribution. For the sake of simplicity, we only introduce the detailed learning behavior of four categories of students in school A.

The engagement of “MOOC-disinterested” students in online learning tasks is not high, and their completion of video watching is not so satisfied. The viewing time of video 1 to video 11 of these students does not reach the average viewing time level of all the schools; many of the students did not complete the test; they were not enthusiastic to complete the test; this became worse along with the time passed; their persistence of conscious learning was not long. However, the average final grade score of “MOOC-disinterested” students ranked second among the four learning patterns. According to manual analysis afterwards, we found that this type of students mainly came from science and engineering disciplines, and they had an advantage in studying the computer course of “*College Information Technology Foundation.*” Therefore, even if this kind of students completed the online learning tasks casually, they can still get good grades.

“Self-motivated” students had longer video watching time and higher test scores. They were self-driven in learning from the beginning of the course and persisted until the end of the course. This learning pattern is the most ideal. These students can finish watching the course videos as requested by them. After learning the content of each part, they can also complete the after-class quizzes in time, to review what

TABLE 1: Number of students for different class in schools.

School	Class 1	Class 2	Class 3	Class 4
School A	38	375	1272	745
School B	403	1061	1346	297
School C	496	1555	1285	276

TABLE 2: Four typical learning modes.

Category	Learning mode	Proportion (%)	Average final grade
Cluster 0	MOOC-disinterested	25	66.7
Cluster 1	Self-motivated	30	70.1
Cluster 2	Video-preferred	31	62.6
Cluster 3	Task-oriented	14	53.3

they had learned. The highest average final score of this type of student is 70.1 points.

“Video-preferred” students had a good performance in watching videos, and almost all the 11 teaching videos can be watched completely. Most of the students submitted the test in December 2016, and some of them submitted the test in January 2017 (the deadline of MOOC course). Although the number of people who did not submit the test is lower in the beginning, the number increased along with the MOOC proceed, which shows that their motivation to complete the test had decreased over time. In addition, the average test score of these students is failed. It can be seen from these phenomena that this type of student had better learning habits, are able to take the initiative to watch the course videos, and can complete the test in time. But they may not pay attention to the after-class quizzes and regard completing the test as a mandatory task. As a result, their score is the third in four patterns. Therefore, the average final grade score of such students is just over the passing line, which is 62.6 points.

The video viewing time, the number of tasks completed, and the average test score of the “task-oriented” students ranked first. The test submission time of this type of student was mainly limited in January 2017, which was the deadline for submission. Most of these students did not complete the test in time to consolidate their knowledge. They completed the test to meet the teacher’s request only. The average final exam score was 53.3. This indicates that this type of student completes online learning tasks in a “task-oriented” style, so the learning effect was poor, and the average final grade of them failed to pass.

The majority of “task-oriented” students cannot pass the final exam, so teachers needed to remind and supervise these students so that they can watch the teaching videos in time and complete homework to help them pass the final exam. Furthermore, in the teaching of the new semester, the current result can be used to carry out corresponding teaching interventions to various types of students, thereby improving the quality of teaching and learning.

4.1. Comparison of Video Watching Behavior of Four Classes.

The average percentage of each video viewing time of the four types of students is shown in Figure 1. Figure 1 shows that the viewing time of the “MOOC-disinterested” students is different from the other three types significantly, while the viewing time of the three clusters of “self-motivated,” “video-preferred,” and “task-oriented” is similar. Ranking all clusters by descending order of video viewing time, “task-oriented” is the first, while “self-motivated” and “video-preferred” follow, and “MOOC-disinterested” is at the bottom. Most of the “MOOC-disinterested” students did not finish watching all videos, while the viewing time of other types basically exceeded 1 in each video, which means these three types of students watched all the videos, and some of them watched the videos repeatedly. However the students in the cluster of “MOOC-disinterested” were not willing to watch the videos, and nearly all the course videos were not finished.

The up and down trend of the four kinds of students were relatively consistent in video viewing curves. All students watched videos 3, 8, and 9 for a long time. This may due to the fact that the content of these three teaching videos involves algorithms, database technology, and basic computer structure. It is difficult to learn and students need to be watched repeatedly.

4.2. Comparison of Quiz Behavior of Four Classes.

The average scores of the four kinds of students in tests 1–8 are shown in Figure 2. Figure 2 shows that the gap among the four groups in test scores is more obvious than in the video viewing behavior. The test scores of “self-motivated” and “task-oriented” are higher, and the scores of each test are relatively close, while the test scores of the “MOOC-disinterested” and “video-preferred” are significantly different from the “self-motivated” and “task-oriented.” The ranking of test scores from top to bottom is “task-oriented,” “self-motivated,” “video-preferred,” and “MOOC-disinterested.”

The scores of “MOOC-disinterested” students are the lowest in each test, and the scores show a decreasing trend. This shows that the “MOOC-disinterested” students were less and less motivated to complete the test over time, and their continuity of self-driven learning is not long. The test scores of “video-preferred” students are higher than those of “MOOC-disinterested” students and rank third place. The fluctuation of each test is small, but they failed in each test. This shows that although “video-preferred” students are able to persist in completing the test, the quality of completion was not high. The scores of the “self-motivated” and “task-oriented” students in each test are relatively close, reaching above 80 points. The “task-oriented” group is slightly higher than the “self-motivated” group, and the test scores of these two types of students show an increasing trend. It indicates that the two kinds of students can persist in completing the test, and the quality of completing the test is high.

Figures 3–6 further compared the time taken by four students to submit tests.

Among the “MOOC-disinterested” students, the number of students who submitted the tests (test 1 to test 8) early

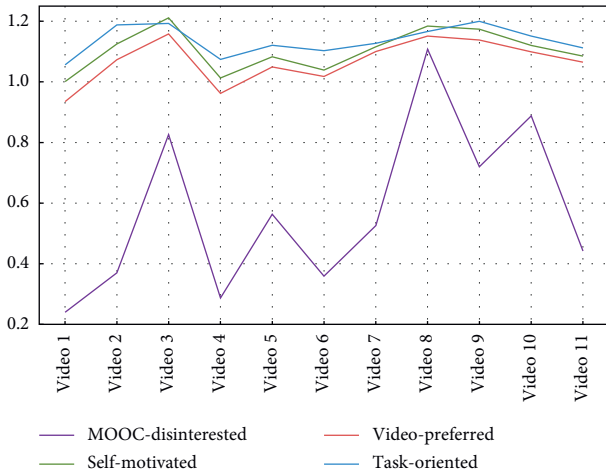


FIGURE 1: Percentage of video viewing time of four clusters (1.0 means that the video has been watched in 100%).

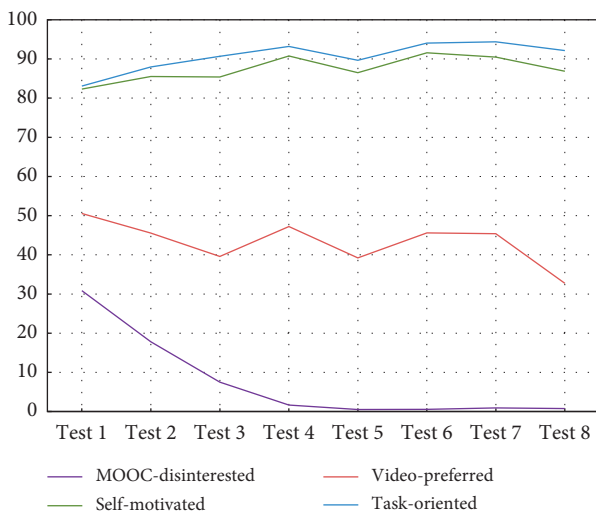


FIGURE 2: Test scores for each of the four clusters.

is quite low and it decreases in later tests, while the number of students who did not submit tests is large and gradually increases. This verifies the assumption above that the enthusiasm of “MOOC-disinterested” students to complete the test is lower and lower over time, and the continuity of self-driven learning was not long.

It can be seen from Figure 4 that, in the “self-motivated,” a large number of students submitted the test in December 2016. These students reviewed and tested the knowledge they learn in time after completing the corresponding course. This shows that they have a good learning attitude and high self-consciousness. From test 1 to test 8, the number of students submitted in each period is basically the same (the number of people who submitted the test in November 2016 decreases slightly, which is related to the curriculum arrangement of course, which means the dates of the test). In addition, “self-motivated” students completed all tests, which shows that such students are more self-driven and their learning enthusiasm did not fade over time.

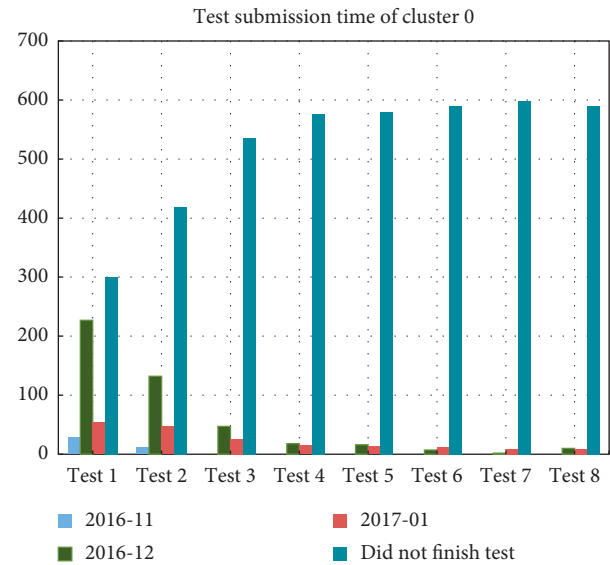


FIGURE 3: Test submission time for “MOOC-disinterested” students.

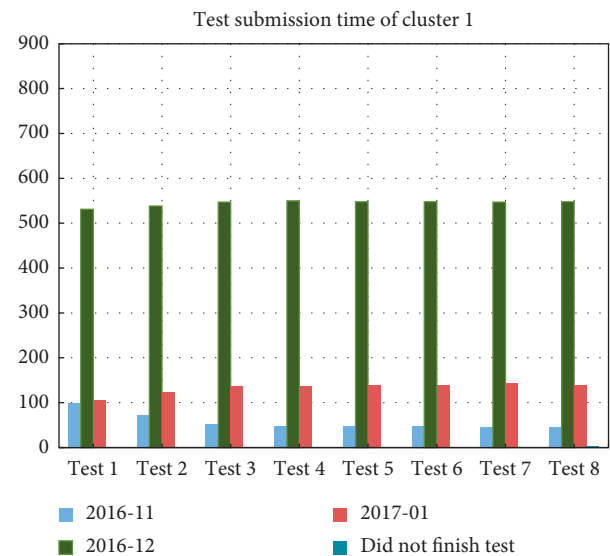


FIGURE 4: Test submission time for “self-motivated” students.

Figure 5 shows the test submission time of “video-preferred” students. Among the “video-preferred” students, the majority of them submitted the test in December 2016, but not as many as the “self-motivated.” Besides, the number of students who submitted in January 2017 is quite large, and the proportion is greater than the “self-motivated” significantly. It also can be seen that the number of students who had not completed the test in the “video-preferred” is more than that of the “self-motivated,” but much less than the “MOOC-disinterested.” This means that although the “video-preferred” students are not as motivated as the “self-motivated” students, most of them had completed the tests and their study habits are better.

Figure 6 reflects the submission time of the “task-oriented” students’ tests. It can be seen that the vast majority of

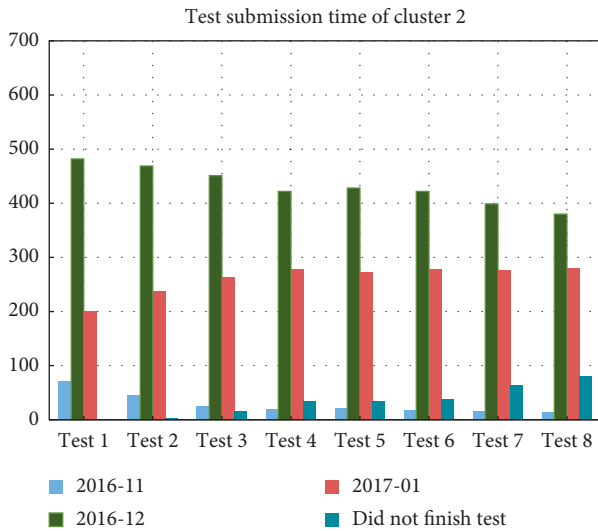


FIGURE 5: Test submission time for “video-preferred” students.

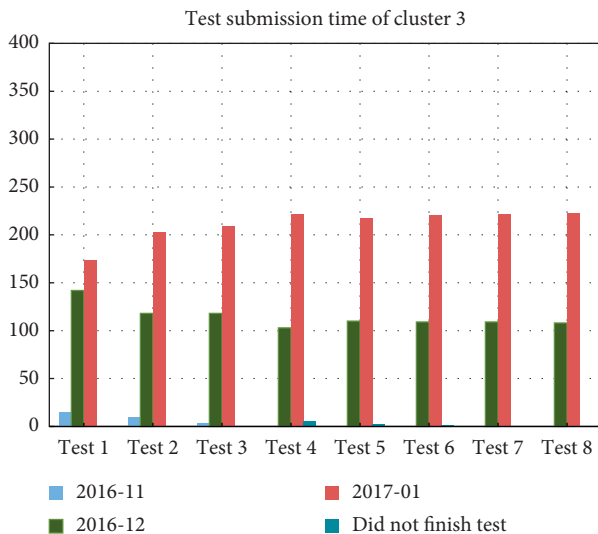


FIGURE 6: Test submission time for “task-oriented” students.

students had submitted the tests, and only a few had not submitted the tests. The completion of the online tests is relatively high, but in the latter part of the course, the number of students who submitted in January 2017 accounts for the largest proportion, and from test 1 to test 8, the number of students who submitted tests gradually decreases in December 2016, and the number of students who submitted tests gradually increases in January 2017. This shows that the “task-oriented” students can basically complete the test but not in time. Most of them finished the test before the deadline for submitting the tests on the MOOC platform. The enthusiasm and consciousness of learning need to be improved. Although the average score of the “task-oriented” students’ test ranks the top of the four types, this is not a learning habit worth to be promoted. It can be assumed that “task-oriented” students complete the tests just to meet the teacher’s request.

By the analysis of Figures 4–6, it can be found that the learning habit of “self-motivated” students’ is the best, and almost all of these students had submitted the tests. The learning habit of “video-preferred” students is the second, and a small part of them had not completed the tests. Most of these two types of students can review and test their knowledge in time after completing the corresponding courses. The students of the “task-oriented” had a corresponding high degree of completion of the online test, and almost all of these students had submitted all the tests. However, most of them submitted the tests in a rush when the deadline approached, and the self-driven learning of them is not as good as the “self-motivated.” Their main purpose of completing the tests may be meeting the teacher’s request only. The completion of “MOOC-disinterested” students is the worst in tests. Many students had not completed all the tests, the enthusiasm for completing the tests got lower and lower over time, and the continuity of self-driven learning does not last long. These students may not care whether they can pass the course of “*College Information Technology Fundamentals*.” Also, they were not very enthusiastic in online learning.

4.3. Comparison of Visiting Behavior of Four Classes.

Figure 7 shows the average visit of four kinds of students. The horizontal axis indicates four types of students ranked by the average visits. And the vertical axis represents the number of average visits. The number of the “task-oriented” is significantly higher than that of the others. Among the other three types, the total number of visits is ranked as follows: “video-preferred,” “self-motivated,” and “MOOC-disinterested.”

5. Grade Prediction by ECOC-Based Classification Algorithms

5.1. ECOC Algorithms. There are two common categories of machine learning algorithms which have been widely used: classification and clustering. Classification can be divided into binary-class and multiclass classification problems. For binary-class problems, many excellent machine learning algorithms have been developed, while for multiclass classification problems, most of the data set cannot be fit effectively by a single classification algorithm; therefore, “partition” is adopted. The “partition” decomposes the multiclass classification problem into several binary-class problems and then uses the base classifiers to perform binary-class classification. This helps to solve the multiclass classification problem effectively.

ECOC algorithm was developed for solving multiclass classification problems, which has been successfully applied in different fields, such as face recognition [21] and bioinformatics [22–24]. It comes from the signal transmission problem in communication. The process of the ECOC algorithm can be divided into three basic steps: encoding, training, and decoding [25]. The key process of the ECOC algorithm to deal with multiclass classification problems lies in the selection of matrix encoding method and decoding

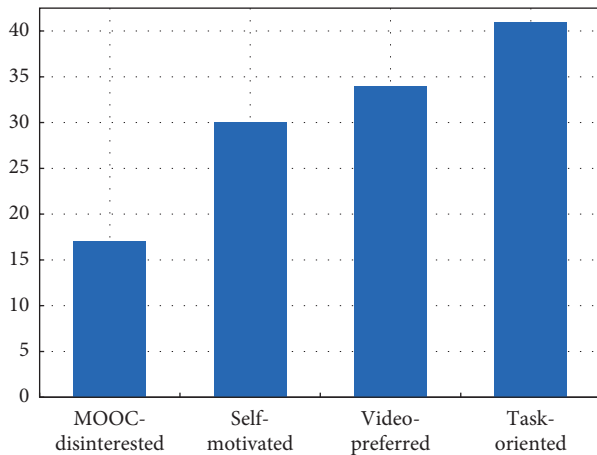


FIGURE 7: Average visits of four types of students.

method. Encoding matrix algorithms can be mainly divided into two categories: data-independent and data-dependent. Data-independent encoding algorithms do not rely on data samples in data encoding. There are mainly One VS One (OVO), One VS All (OVA) [26], Dense Random (DR) [27], and Sparse Random (SR) [28] encoding matrix. Data-dependent encoding algorithms have different encoding matrices for different data sample distribution. Such encoding methods include DECOC, Forest-ECOC, and ECOC-ONE [29]. The ternary coding strategy is widely used in ECOC, where each element can take a value in the set $\{0, +1, -1\}$. At this time, $+1$ or -1 denotes that the original class is assigned to the positive class group or the negative class group, respectively; 0 denotes that there is no corresponding class in the training process, which means the class is ignored.

5.2. GA-Based Feature Selection Methods. In general, there are some redundant or irrelevant attributes (features) in the original data set. In order to avoid the curse of the dimensionality and improve the classification accuracy of the model as well, feature selection is considered as an effective method to pick up most representative subset of features from the data set. In this paper, GA is employed to realize the feature selection method because it has powerful searching ability [30] and been successfully applied in feature selection [31, 32]. Thus, GA is developed to construct an efficient ECOC ensemble system with feature selection.

5.3. The Design of GA-ECOC Classification Method. This paper proposes a multiclass classification algorithm framework based on GA and ECOC for student grade prediction, namely, GA-ECOC. In the framework, GA is used as a feature selection method. The feature of data set is encoded by binary chromosome. Each bit of the chromosome represents a feature: 1: the feature is selected, and 0: the feature is not selected. The school A data set has 35 fields, school B has 82 fields, and school C has 153 fields. For the samples from 3 schools, the remaining fields are used as the characteristics except the final grade of each sample. The GA has two operations: crossover operation and mutation

operation. The crossover operation is selecting two individuals randomly from the population and randomly selecting one of the chromosomal positions as the crossover position and crossover to obtain two new individuals. The mutation operation is selecting a bit randomly from each individual to mutate according to a certain probability. The average classification accuracy of ECOC is used as the fitness function of GA. And the method sorts the population individuals according to their fitness; the top 20% of individuals with high fitness are selected firstly. Based on the random selection probability, the remaining individuals having a 50% probability are selected to produce the next generation. In order to verify the effectiveness of the proposed approach, tenfold cross-validation is employed for all experiments; the data set is divided into 10 folds; one of the folds is used for testing and others are used as training data in each interaction. In the experiment, SVM classifier is used as the base classifier in ECOC, and the average accuracy of all interactions is used as the individual fitness. The framework of the methodology is shown in Figure 8.

6. Experiments and Discussions

This paper uses the learning behavior data of three schools to predict students' grade. The parameters of the experiment are set as follows: the initial population of the GA is 100, the number of iterations is 50 generations, the mutation rate is 0.01, the optimize selection rate is 0.2, and the random selection rate of the remaining individuals is 0.5. SVM is selected as the base classifier of ECOC, according to the average accuracy of 10-fold cross-validation. In the individual chromosomes, the length of the individual chromosomes algorithm of school A is 34 bits corresponding to the algorithm, while the length of school B is 81 bits and the length of school C is 152 bits. And the ensemble learning algorithm, Random Forest [33], is used for comparison.

6.1. Performance Comparison of Methods. Firstly, the prediction result of the ECOC and Random Forest algorithm is shown in Table 3. It can be seen from that, for the data set of school A, the average accuracy obtained by the ECOC algorithm was about 79%. For school B, the average accuracy obtained by the ECOC algorithm was approximately 70%, and for school C, the average accuracy is approximately 68%. What is more, all the ECOC algorithms can perform better than the Random Forest in three data sets.

In further experiments, the result of the multiclass classification algorithm (GA-ECOC) is shown in Table 4. The result can be analyzed in two aspects, accuracy and encoding methods. Firstly, for accuracy, the average accuracy of school A is about 80%, while the average accuracy of school B is about 75% and the average accuracy of school C is about 74%. Secondly, for the encoding methods used in three data sets, the SR encoding method achieves the highest accuracy, followed by the DR encoding. OVA encoding and OVO encoding have poor performance in accuracy, compared with the two methods above. Finally, the result of the data set from school B and school C shows that the accuracy

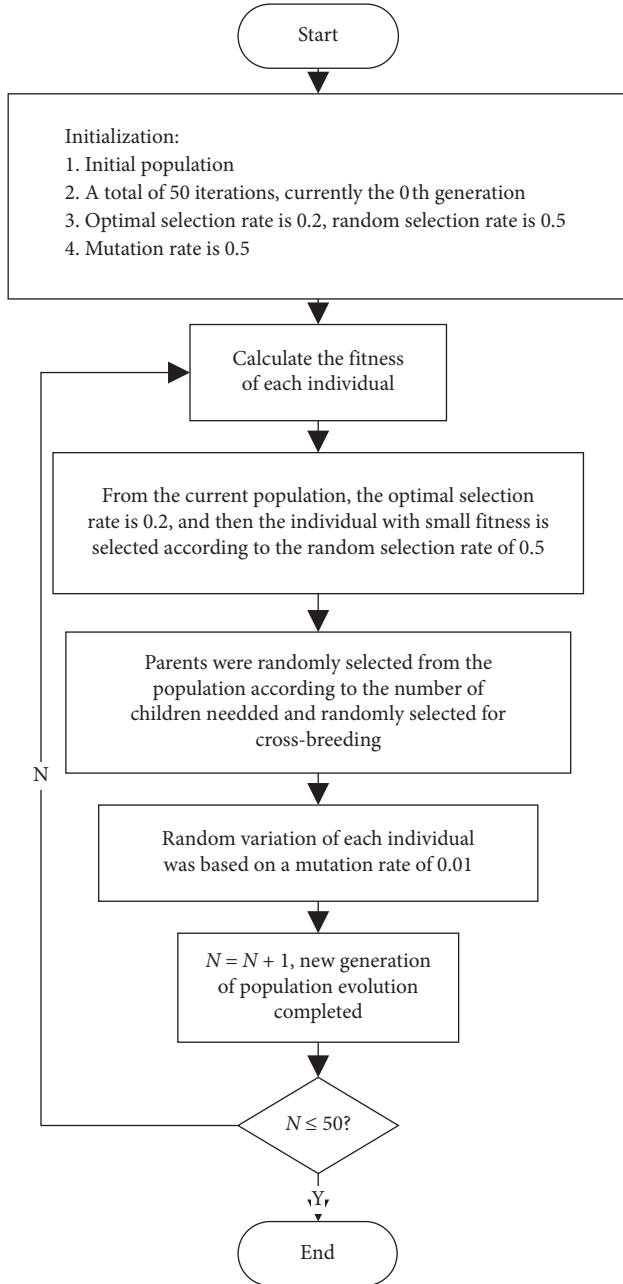


FIGURE 8: Flowchart of multiclass classification algorithm (GA-ECOC).

rate of school B is about 5% lower than that of school A, and the accuracy rate of school C is about 6% lower than that of school A. The reason may be that the feature dimension of school B data is twice as the dimension of school A, the feature dimension of school C is even higher, schools B and C have more space for optimization than school A, and therefore the classification accuracy of the model has a significant decrease.

The results of the two algorithms can be compared in three data sets. It can be seen from Figure 9 that, for school A data set, the accuracy rate obtained by the GA-ECOC algorithm is about 2% higher than those obtained by the ECOC algorithm. Figure 10 indicates that, for the school B

TABLE 3: Average accuracy of the Random Forest and ECOC algorithms for grade prediction.

Data sets	Random Forest	OVO	OVA	DR	SR
School A	0.676	0.747	0.805	0.804	0.806
School B	0.436	0.706	0.704	0.698	0.703
School C	0.406	0.689	0.676	0.670	0.692

TABLE 4: Average accuracy of GA-ECOC algorithms for grade prediction.

Data sets	OVO	OVA	DR	SR
School A	0.776	0.819	0.821	0.822
School B	0.756	0.749	0.751	0.757
School C	0.745	0.735	0.743	0.748

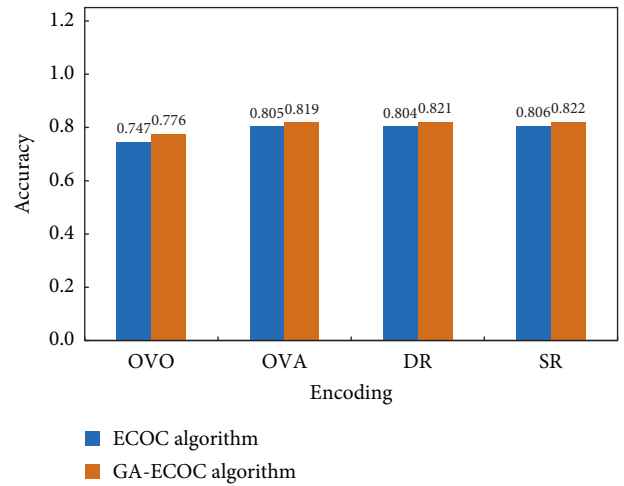


FIGURE 9: Performance comparison between ECOC and GA-ECOC algorithms on school A.

data set, the accuracy of the GA-ECOC algorithm is improved by about 5%, compared with the ECOC algorithm. Figure 11 indicates that, for school C data set, the accuracy of the GA-ECOC algorithm is improved by about 6% on average, compared to the ECOC algorithm.

Based on the experiments above, the classification performance of GA-ECOC algorithms is significantly better than the ECOC algorithm in all data sets. The main reason may be that some features in the SPOC data are not related to student performance or redundant, and some features have a strong correlation with student performance. GA is able to select an effective feature subset, which helps to improve ECOC Classification accuracy. Therefore, feature selection can not only improve the classification accuracy effectively with four different ECOC matrix encoding methods (OVO, OVA, DR, and SR) but also improve the classification accuracy significantly in different school data set.

6.2. Analysis of Computational Complexity of Methods. Table 5 shows the comparison of the time complexity of different algorithms. The training time of the GA-ECOC

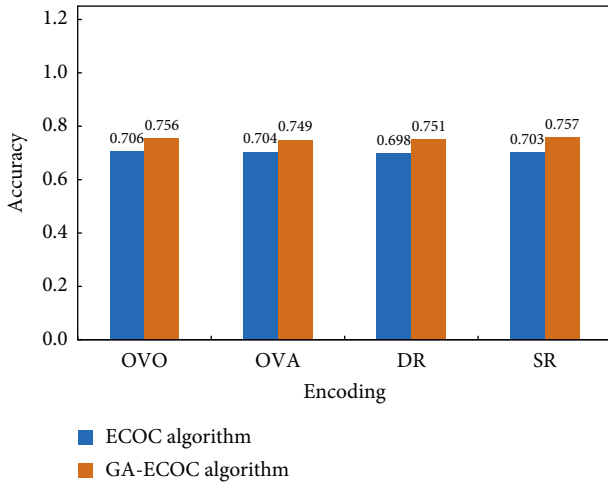


FIGURE 10: Performance comparison between ECOC and GA-ECOC algorithms on school B.

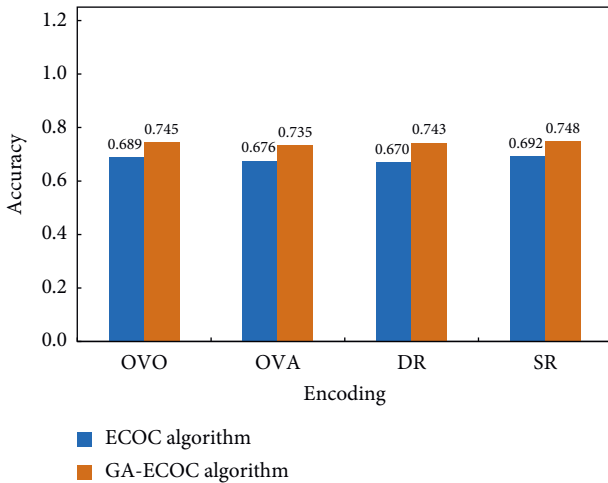


FIGURE 11: Performance comparison between ECOC and GA-ECOC algorithms on school C.

TABLE 5: Comparison of running times of algorithms (in seconds).

Methods	School A	School B	School C
ECOC	23	40	84
GA-ECOC-OVO	15,885	41,635	86,677
GA-ECOC-OVA	29,539	77,410	168,265
GA-ECOC-DR	48,641	133,339	280,350
GA-ECOC-SR	190,256	389,493	683,749

algorithm is much longer than the ECOC algorithm. In the data sets of the three schools, the ECOC algorithm is used with different codes. They only use less than 90 seconds to complete the training process, which means that the algorithm complexity is lower. After introducing GA as a feature selection method, multi-iteration is conducted, which is computationally expensive, to find the optimal feature subset. Most of the algorithms required more than 4 hours to complete model training. What is more, the complexity of GA calculation is different from data sets with different feature numbers. School C has 152 features, the individual

chromosome is the longest among three schools, the feature dimension to be searched is quite large, and the computing time is the longest; school B has 81 features, which are fewer than those in school C, the feature dimension to be searched is small, so the computing time is less. The data of school A has only 34 features. With the fewest features, the computing time is the shortest. It takes around 4 hours only to complete feature selection and model training. On the other hand, different ECOC encoding methods have different calculation costs. OVA and OVO matrix coding are basic relatively, and the corresponding matrix generation method is simple; while DR and SR are the method with random coding, the matrix generation method is complicated, and it is easy to cause coding problems, which have to debug and rerun until a satisfactory result is acquired, so more time is required. Therefore, the calculation time of GA-ECOC based on DR and SR is longer than that based on OVA and OVO.

7. Conclusion

With the rise of SPOC blended teaching, using student online/offline learning behavior log data to analyze and predict students' learning performance is an important factor for personalized education. This paper analyzes the learning behavior of students in SPOC blended teaching in three schools by the clustering algorithm, ECOC algorithm, and GA-ECOC algorithm. The main findings are shown as follows:

- (1) By the clustering algorithm, combining with the teaching experience accumulated by staff, typical learning patterns of students can be classified into four clusters. The learning behavior patterns among the four clusters are significantly different. Learning the patterns is quite helpful in curriculum design and providing personalized guidance for students.
- (2) A multiclass classification algorithm based on GA and ECOC is designed, in order to predict student performance. The powerful searching capability of GA in feature selection and then ECOC algorithm makes a good performance in multiclass prediction. Compared with the Random Forest and single ECOC algorithm, the classification prediction of GA-ECOC algorithms is more precise and effective.

The next-stage work can be considered from the following aspects:

- (1) For data collection, more teaching-related data should be collected, such as students' previous academic performance and students' offline learning behavior in order to have a full picture of students' learning process.
- (2) The algorithm should be designed to identify different learning behavior patterns and predict students with different learning behavior patterns in advance. This can help to provide different suggestions to the staff, according to different students; therefore, the teaching quality and student learning experience can be improved.

- (3) For the design of the multiclass classification algorithm, the types of the coding matrix generated are limited by the method of the traditional coding matrix, and the best classification performance cannot be achieved. Therefore, the searching method of feature subset and method of coding matrix generation should be optimized, and this will become an essential factor in future research.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (no. 61772023), Key Research and Development Program of China (no. 2019QY1803), Natural Science Foundation of Fujian Province of China (nos. 2020J01697, 2020J01707, 2020R0066, and 2018J01538), and the Opening Fund of Digital Fujian Big Data Modeling and Intelligent Computing Institute.

References

- [1] D. T. Seaton, Y. Bergner, I. Chuang, P. Mitros, and D. E. Pritchard, "Who does what in a massive open online course?" *Communications of the ACM*, vol. 57, no. 4, pp. 58–65, 2014.
- [2] D. O. Bruff, D. H. Fisher, K. E. McEwen, and B. E. Smith, "Wrapping a MOOC: student perceptions of an experiment in blended learning," *Journal of Online Learning and Teaching*, vol. 9, no. 2, p. 187, 2013.
- [3] J. Kay, P. Reimann, E. Diebold, and B. Kummerfeld, "MOOCs: so many learners, so much potential," *IEEE Intelligent Systems*, vol. 28, no. 3, pp. 70–77, 2013.
- [4] J. Gardner and C. Brooks, "Student success prediction in MOOCs," *User Modeling and User-Adapted Interaction*, vol. 28, no. 2, pp. 127–203, 2018.
- [5] S. G. Wilson, "The flipped class: a method to address the challenges of an undergraduate statistics course," *Teaching of Psychology*, vol. 40, no. 3, Article ID 0098628313487461, 2013.
- [6] D. Wagner, P. Laforge, and D. Cripps, "Lecture material retention: a first trial report on flipped classroom strategies in electronic systems engineering at the University of Regina," in *Proceedings of the Canadian Engineering Education Association*, Montreal, Canada, June 2013.
- [7] V. Ng, R. L. Huang, L.-C. Hong, and K. Liu, "Are my students ready: a case of flipped learning in an it subject," in *Proceedings of the 11th International Conference on E-Learning*, Kuala Lumpur, Malaysia, June 2016.
- [8] A. Anderson, D. Huttenlocher, J. Kleinberg, and J. Leskovec, "Engaging with massive online courses," in *Proceedings of the 23rd International Conference on World Wide Web*, pp. 687–698, Seoul, South Korea, April 2014.
- [9] G. Hughes and C. Dobbins, "The utilization of data analysis techniques in predicting student performance in massive open online courses (MOOCs)," *Research and Practice in Technology Enhanced Learning*, vol. 10, no. 1, pp. 10–18, 2015.
- [10] J. Qiu, T. Jie, T. X. Liu, G. Jie, and Y. Xue, "Modeling and predicting learning behavior in MOOCs," in *Proceedings of the Ninth ACM International Conference on WSDM'16*, San Francisco, CA, USA, February 2016.
- [11] J. Leskovec, D. Huttenlocher, and J. Kleinberg, "Predicting positive and negative links in online social networks," in *Proceedings of the 19th International Conference on World Wide Web*, pp. 641–650, Raleigh, NC, USA, April 2010.
- [12] B. Xu and D. Yang, "Motivation classification and grade prediction for MOOCs learners," *Computational Intelligence and Neuroscience*, vol. 2016, Article ID 2174613, 7 pages, 2016.
- [13] J. Bailey, R. Zhang, B. Rubinstein, and J. He, "Identifying at-risk students in massive open online courses," in *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, pp. 1749–1755, Austin, TX, USA, January 2015.
- [14] A. Ezen-Can, K. E. Boyer, S. Kellogg, and S. Booth, "Unsupervised modeling for understanding MOOC discussion forums: a learning analytics approach," in *Proceedings of the Fifth International Conference LAK '15*, pp. 146–150, Poughkeepsie, NY, USA, March 2015.
- [15] M. Zhang, J. Zhu, Z. Wang, and Y. Chen, "Providing personalized learning guidance in MOOCs by multi-source data analysis," *World Wide Web*, vol. 22, pp. 1189–1219, 2019.
- [16] C.-H. Yu, J. Wu, and A.-C. Liu, "Predicting learning outcomes with MOOC clickstreams," *Education Sciences*, vol. 9, no. 2, p. 104, 2019.
- [17] Y. Meier, J. Xu, O. Atan, and M. van der Schaar, "Predicting grades," *IEEE Transactions on Signal Processing*, vol. 64, no. 4, pp. 959–972, 2016.
- [18] J. Xu, K. H. Moon, and M. van der Schaar, "A machine learning approach for tracking and predicting student performance in degree programs," *IEEE Journal of Selected Topics in Signal Processing*, vol. 11, no. 5, pp. 742–753, 2017.
- [19] R. L. Ulloa-Cazarez, C. López-Martín, A. Abran, and C. Yáñez-Márquez, "Prediction of online students performance by means of genetic programming," *Applied Artificial Intelligence*, vol. 32, no. 9-10, pp. 858–881, 2018.
- [20] M. Hussain, W. Zhu, W. Zhang, S. M. R. Abidi, and S. Ali, "Using machine learning to predict student difficulties from learning session data," *Artificial Intelligence Review*, vol. 52, no. 1, pp. 381–407, 2019.
- [21] S. Nazari, M. S. Moin, and H. Rashidy Kanan, "Securing templates in a face recognition system using error-correcting output code and chaos theory," *Computers & Electrical Engineering*, vol. 72, pp. 644–659, 2018.
- [22] M. Sun, K. Liu, Q. Wu, Q. Hong, B. Wang, and H. Zhang, "A novel ECOC algorithm for multiclass microarray data classification based on data complexity analysis," *Pattern Recognition*, vol. 90, pp. 346–362, 2019.
- [23] Y. P. Zhang, X. N. Ye, K. H. Liu, and J. F. Yao, "A novel multi-objective genetic algorithm based error correcting output codes," *Swarm and Evolutionary Computation*, vol. 57, Article ID 100709, 2020.
- [24] X.-N. Ye, K.-H. Liu, and S.-T. Liong, "A ternary bitwise calculator based genetic algorithm for improving error correcting output codes," *Information Sciences*, vol. 537, pp. 485–510, 2020.
- [25] T. G. Dietterich and G. Bakiri, "Solving multiclass learning problems via ECOC," *Journal of Artificial Intelligence Research*, vol. 2, no. 1, pp. 263–286, 1994.
- [26] M. Galar, A. Fernández, E. Barrenechea, H. Bustince, and F. Herrera, "NMC: nearest matrix classification—a new

- combination model for pruning one-vs-one ensembles by transforming the aggregation problem,” *Information Fusion*, vol. 36, pp. 26–51, 2017.
- [27] K. Crammer and Y. Singer, “On the learnability and design of output codes for multiclass problems,” *Machine Learning*, vol. 47, no. 2-3, pp. 201–233, 2002.
- [28] L. Zhou, Q. Wang, and H. Fujita, “One versus one multi-class classification fusion using optimizing decision directed acyclic graph for predicting listing status of companies,” *Information Fusion*, vol. 36, pp. 80–89, 2017.
- [29] F. Masulli and G. Valentini, “Effectiveness of error correcting output coding methods in ensemble and monolithic learning machines,” *Formal Pattern Analysis & Applications*, vol. 6, no. 4, pp. 285–300, 2004.
- [30] D. Dutta, J. Sil, and P. Dutta, “Automatic clustering by multi-objective genetic algorithm with numeric and categorical features,” *Expert Systems with Applications*, vol. 137, pp. 357–379, 2019.
- [31] L. I. Kuncheva and L. C. Jain, “Designing classifier fusion systems by genetic algorithms,” *IEEE Transactions on Evolutionary Computation*, vol. 4, no. 4, pp. 327–336, 2000.
- [32] M. A. Bagheri, Q. Gao, and S. Escalera, “A genetic-based subspace analysis method for improving error-correcting output coding,” *Pattern Recognition*, vol. 46, no. 10, pp. 2830–2839, 2013.
- [33] L. Breiman, “Random forests,” *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.