Hindawi
Mathematical Problems in Engineering
Volume 2022, Article ID 9451108, 17 pages
https://doi.org/10.1155/2022/9451108

Research Article

Autonomous Classification and Decision-Making Support of Citizen E-Petitions Based on Bi-LSTM-CNN

Fengmei Sun[1] **and Yi Zuo**[2,3]

[1]*Collaborative Innovation Center for Transport Studies, Dalian Maritime University, Dalian 116026, China*
[2]*Collaborative Innovation Center of Maritime Big Data & Shipping Artificial General Intelligence, Navigation College Dalian Maritime University, Dalian 116026, China*
[3]*The Research Institute for Socionetwork Strategies, Kansai University, Osaka 5648680, Japan*

Correspondence should be addressed to Yi Zuo; zuo@dlmu.edu.cn

Received 27 May 2022; Revised 12 August 2022; Accepted 22 August 2022; Published 16 September 2022

Academic Editor: Toqeer Mahmood

The increasing number of e-petition services requires accurate calculation methods to perform rapid and automated delivery. Automated text classification significantly reduces the burden of manual sorting, improving service efficiency. Moreover, existing text classification methods focus on improving sole models with an insufficient exploration of hybrid models. Moreover, existing research lacks combinatorial model selection schemes that yield satisfactory performance for petition classification applications. To address these issues, we propose a hybrid deep-learning classification model that can accurately classify the responsible department of a petition. First, e-petitions were collected from the Chinese bulletin board system and then cleaned, segmented, and tokenized into a sequence of words. Second, we employed the word2vec model to pretrain an embedding matrix based on the e-petition corpus. An embedding matrix maps words into vectors. Finally, a hybridized classifier based on convolutional neural networks (CNN) and bidirectional long short-term memory (Bi-LSTM) is proposed to extract features from the title and body of the petition. Compared with baseline models such as CNN, Bi-LSTM, and Bi-LSTM-CNN, the weighted $F1$ score of the proposed model is improved by 5.82%, 4.31%, and 1.58%, respectively. Furthermore, the proposed automated petition classification decision support system is available on the e-petition website and can be used to accurately deliver petitions and conduct citizen opinion analysis.

1. Introduction

Online applications of e-government have been increasingly applied to pursuing citizen demand and government regulations that allow citizens to interact efficiently with government agencies. One of the most widely used e-government applications is e-petition, which enhances the communication of citizens with functional departments [1]. Recently, the burden of e-petition delivery has grown, and the number of messages has significantly increased from hundreds per week to thousands per day for government offices [2, 3]. According to the Shanghai Municipal Bureau of Petition in China, over 69,000 public opinions and suggestions were received and handled in 2021, with a monthly average of 5,750 cases and an average public response rate of 79.08% in each district. Also, 49,554 petitions have been received from January to April 2022, with the number of petition submissions increasing rapidly and the daily response rate dropping to below 70%.

The government requires the ability to automatically determine the content of these petitions to reduce this burden. Alternatively, the exponential growth of easily accessible text data has led to a surge of research interest in automated content methods, such as automated text classification (ATC) [4, 5]. ATC is a subfield of natural language processing (NLP) that assigns documents to one or more prespecified categories, while these texts are usually unstructured [6, 7]. Large amounts of e-petition data are

publicly available. These data are novel in their availability and high storage volume, not widely used for research purposes [8]. ATC might offer an effective solution to address automated e-petition deliveries. However, selecting an accurate and efficient model is a challenge for current applications [9].

In traditional text classification research, classical classifiers include support vector machines (SVM) [10, 11], k-nearest neighbor (KNN) [12], and naive Bayes (NB) [13]. However, these methods cannot extract contextual relationships between words and text [5]. Since the concept of deep learning was proposed in 2006 [14], Kim [15] first proposed a text-based convolutional neural network (TextCNN). The model performance was verified on seven short-text data sets with an average length of below 25. The outstanding performance of convolutional neural network (CNN) in short-text classification is attributed to the local correlation captured by the convolution and pooling operations. The limitation of CNN lies in the loss of structural information in maximum pooling; therefore, finding complex patterns such as adversative relations in long texts is difficult. Subsequently, a recurrent neural network (RNN) was applied to extract global semantic information from texts, which is suitable for time-series analyses such as text classification [16]. Long short-term memory (LSTM) is a special type of RNN. LSTM effectively solves the problems of long-term memory dependence and gradient disappearance in backpropagation and is more effective in processing chapter-level texts [17]. As an improvement of LSTM, bidirectional long short-term memory (Bi-LSTM) has forwarding and backward directions, which excels in extracting context features [18]. Improvements in these text classification models focus more on the network structure, activation function, etc. [19–22]. As deep learning models alone cannot improve classification performance, several scholars have proposed hybridized models [6, 23, 24]. The hybridized model takes advantage of each other to improve the accuracy of text classification. However, no single model is generic for all tasks, and different types of models are useful for specific domains [25]. Modeling unstructured text to achieve automated classification is a challenging task; for instance, a submitted e-petition contains two parts: title and body. Furthermore, the text length varies considerably, so extracting features using a unified method is ineffective.

To address this issue, this study proposes a hybrid autonomous classification method for e-petitions based on Bi-LSTM and CNN and integrates the rich information of the petition text to solve the decision-making support of the autonomous transmission of e-petitions. The proposed hybrid model combines the CNN and LSTM algorithms. The preprocessed title and body of the petition were input in parallel and represented as word embeddings. Then, a CNN with a small convolution kernel captures local features of the concise title, whereas Bi-LSTM-CNN extracts complex semantic features of the complex body. These two features were concatenated to form a unified representation and provide information for classification. To verify the performance of the proposed method, we collected e-petitions from the Chinese bulletin board system (BBS). The numerical results show that the proposed method achieved the highest weighted precision (0.8288) and weighted recall (0.8262). The $F1$ score reached 0.8267, which was higher than that of the baseline models. In addition, we also found that the classification effect of the model using the CNN structure to extract the features of the title was better than that of the model using the LSTM structure. The proposed model can effectively solve the problems of inconsistent text lengths and difficulty in feature extraction. The contributions of this study are twofold. First, it proposes a hybrid deep learning model incorporating title and body features, which is more adaptable to text classification tasks with more information than existing neural network structures. The second is the decision support system, which helps citizens automatically select the responsible department and to provide references for staff to judge petition demands. These two improvements significantly increase the efficiency and effectiveness of the petition workflow.

The remainder of this study is organized as follows. Section 2 reviews related studies on ATC and their applications. In Section 3, we introduce the methodological idea and the proposed classification model. The numerical experiments are presented in Section 4, and we conclude the study in Section 5.

2. Research Background and Literature Review

2.1. Related Studies of Automated Text Classification. Recently, ATC has attracted considerable attention from researchers in NLP. ATC techniques have developed rapidly and profoundly with the support of a large-scale research community. Table 1 lists several advanced studies on text-classification techniques.

Machine learning is considered superior to dictionary-based text classification methods [5]. Traditional machine learning models are widely used in various small-scale text classification tasks, owing to their simple implementation and high interpretability. The corpus size is usually above 10,000, and traditional text representation is high dimensional and highly sparse with weak feature representation, which usually requires additional feature engineering. SVM can effectively pursue two main properties of textual data: high dimensionality and sparsity [11]. However, an SVM requires extensive storage space and training time when the data volume is large [10]. Classic models are adept at solving the problem with strong individual signal words; they may make the classification insufficient because of ignorance of the interaction between words [12, 13].

Deep learning solves the limited representation of complex functions in previous shallow-structured algorithms and has become the mainstream research method in text classification [14]. A CNN has achieved significant success in image classification. The convolutional kernel extracts multidimensional features by setting different weights to obtain local key information through the pooling layer. A CNN enables fast dimensionality reduction and relatively few training parameters through its unique network structure and weight-sharing strategy [28]. Recently, CNN has been applied to various NLP tasks [29] and has

Table 1: Advanced studies on text classification techniques.

Model	Data set	Training set	Classes	Text length	Vocab
SVM [11]	Newspaper	250	2	500	11672
KNN [12]	Newspaper	4731	87	1000	10000
CNN [15]	Customer reviews	12700	3	16	13612
CNN [26]	Twitter	1600000	2	60	76643
LSTM [17]	Newspaper articles	8552	2	339	-
Bi-LSTM [18]	Service documents	9121	10	128	-
CNN + LSTM [6]	Clinical notes	26363	50	200	49354
LSTM-CNN [27]	Newspaper	50000	10	100	-

been extended to Text-CNN, which has several convolutional windows of varied sizes and has an excellent classification effect in the short-text classification task [15, 26]. CNN cannot effectively capture long-term context information between discontinuous words, which is important in text models [30]. To solve this limitation, RNN and LSTM can efficiently explore the potential semantic information of text, and LSTM is more common in long texts [17]. Bi-LSTM has two parallel layers that propagate in both forward and backward directions to fully capture dependencies in the context [18]. However, the accuracy of LSTM is further hampered by the inability to identify the different relationships between the various parts of the document [24]. In addition, CNN can be efficiently computed in parallel, whereas LSTM is less efficient than CNN because of the sequence dependencies problems in transmission [6, 25].

To improve classification, some researchers have proposed hybrid neural networks that combine CNN and LSTM algorithms. The convolution and pooling operations of CNN help the model extract locally salient features, whereas LSTM solves the long-sentence dependency problem by using a gate control mechanism to extract more complex contextual information [6, 31]. Regarding the hybrid model, attention should be paid to the organization of the structure [24]. The model structure design depends on the nature of the problem, and the data researched in the existing literature are derived from news, reviews, and social media. The design scheme of a hybrid model structure explored for novel data sets is rare.

2.2. Related Applications of Text Classification. The prospect of affordable access to substantial amounts of text data has encouraged opportunities for various analytic business, academic, and contextual exploits [5]. Text-classification applications have gained considerable attention from scholars in the field of NLP [27, 32, 33].

Bencke et al. [34] collected messages on citizens' complaints on official social networks regarding transportation, entertainment, and other services. They classified the messages into 14 corresponding dimensions of smart cities. The study compared different feature extraction methods to help classical classification models alleviate the dimensionality problem and pointed out the problem that the classification effect is affected by unbalanced data distribution. [35] proposed an intelligent system to monitor public opinion (i.e., favorable, neutral, and unfavorable) on vaccination decisions, which adopted a bag of words for text representation and achieved the best accuracy with a small sample of training data (693 pieces of data). [36] analyzed law-related news (i.e., relevant, and irrelevant) and proposed a learning method that combines titles and body text, using a bidirectional gated recurrent unit (Bi-GRU) to encode title and body text, and then constructed a bidirectional attention stream to integrate information from both. A new approach for automatically analyzing political discourses of citizens and civil servants was proposed in [22], where they used multiscale CNNs in seven different languages to construct a discourse classifier to classify political manifestos into seven policy domains as well as political party background information to enhance the predictive power of the network.

Table 2 summarizes the text classification intelligent systems that support decision-making for governments. From the perspective of data sources, existing research collected data from specialized social networks [22, 34] and general social networks [35, 36], where general social network information requires additional operations to filter relevant texts. For the sample, unevenly distributed data adds difficulties to the multiclassification task. Methodologically, feature selection and deep learning are used to alleviate the problem of the high sparsity and high dimensionality of textual data. Finally, the text used in the model had a single body part, and the title information was ignored. In relatively few studies that consider titles, uniform structures are used to extract the characteristics of different components, lacking the analysis of textual characteristics. To address these issues, we propose a hybrid model based on deep learning to classify information-rich text. Developing e-government and establishing specialized channels have made accessing substantial amounts of petition data easy. Word2vec was utilized to convert words into low-dimensional vectors to solve the sparse word representation problem. We also analyzed the characteristics of each component of the e-petition text and designed various neural network structures to improve the classification ability of the model.

3. Materials and Methods

The proposed method combines CNN and Bi-LSTM to classify petition text. Figure 1 shows an overview of the proposed method.

3.1. Data Collection. The progress of the petitions and replies were selected and posted on the Shanghai e-petition BBS. Citizens register with their real names, log in, fill out the petition, and submit it. The petitions submitted are received

Table 2: Text classification intelligent systems supporting decision-making for governments.

Function	Data set	Class	Method	Text
Classification of smart city service [35]	Colab	14	Logistic regression	Body
Social media monitor [36]	Tweets related to vaccination	3	SVM + n-gram	Body
Public opinion analysis [22]	Law-related news	2	Bi-GRU	Title + body
Political analysis [37]	Political manifestos	7	CNN + Word2vec	Body

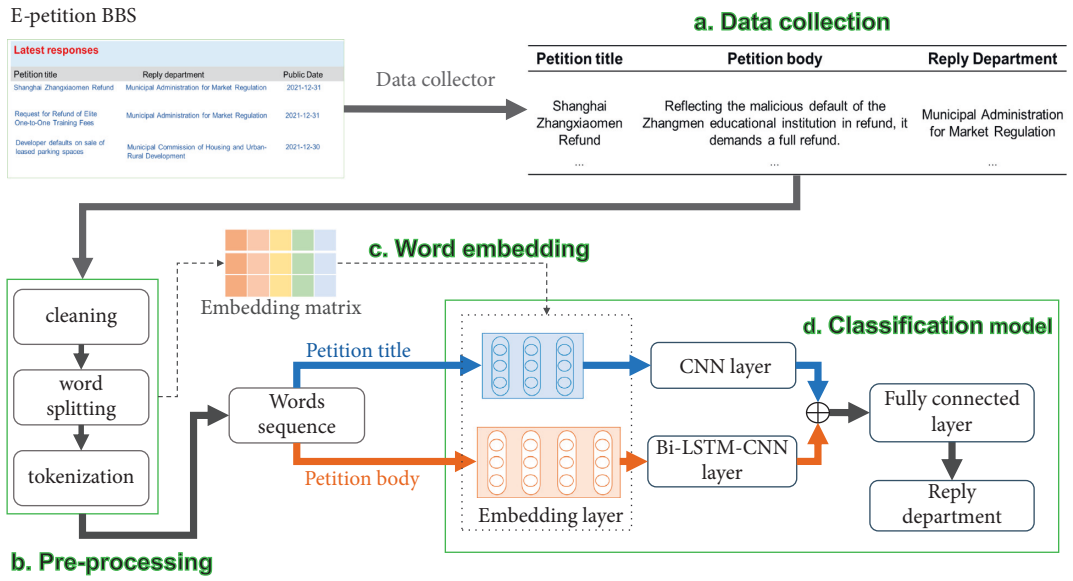

Figure 1: The proposed automated petition classification method overview. (a) Data collection; (b) pre-processing; (c) word embedding; (d) classification model.

and registered regularly by office workers and then forwarded to the relevant functional departments for proper handling according to their duties.

This study used the octopus data collector to automatically crawl petition text data in the BBS. An octopus is an automatic data collection application that simulates human behavior and supports user-designed collection processes. Figure 2 illustrates the collection process for this research. First, the petition bulletin board web page is opened within the collector and contains links to multiple publicized petitions. Clicking on the links leads to a petition details page, and clicking on the corresponding positions on the details page can conveniently capture the text in the corresponding fields on the page. An octopus can generate and loop all links, open the linked pages in turn, and automatically collect the text of the specified fields until all the corresponding texts of the links are collected. The collected data were saved locally in CSV file format. After data collection, each petition text was automatically labeled according to the reply to department. The automatic transmission problem of petition text was transformed into an ATC problem.

3.2. Data Preprocessing. Data preprocessing is the process of transforming raw text into the format required by ML algorithms. We preprocessed the raw data using three operations: cleaning, word splitting, and tokenization. Data preprocessing was implemented using python3 (see

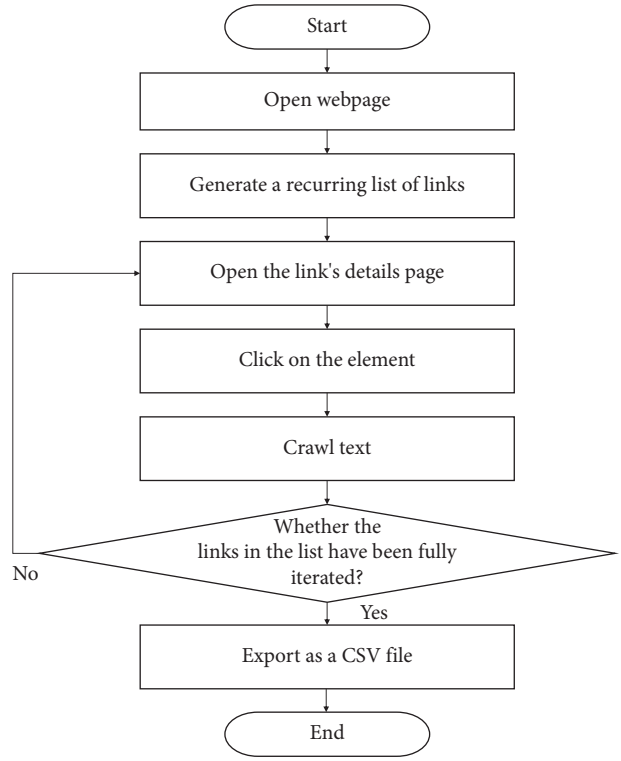

Figure 2: E-petition data collection process with octopus collector: design of the Data Collection Process.

```
Input: texts, user_dictionary_list, stopwords_list
Output: word_sequences
(1)  //Step 1: content cleaning
(2)  m = size of (texts) //get number of total documents in texts
(3)  For i = 1 to m do
(4)     Content = texts[i] //get the content of i_th document
(5)  Content = content.strip() //remove the blanks
(6)  n = length (content) //get number of total characters in content
(7)     For j = 1 to n do
(8)        Character = content[j]
(9)        If character < u'\u4e00' and character > u'\u9fa5' then
(10)          Character = character.strip() //remove the blanks
(11)       End if
(12)    End for
(13) End for
(14) //Step 2: word splitting
(15) Jieba.load () //jieba is a python Chinese word segmentation module
(16) User_dictionary_list.load ()
(17) Stopwords_list.load ()
(18) Corpus = [] //build internal corpus
(19) For i = 1 to m do
(20)    Word_list = jieba.lcut (texts[i]) //split texts[i]
(21)    s = length (word_list) //get number of total words in word_list
(22)    For j = 1 to s do
(23)       If word_list[j] in stopwords_list then
(24)          Remove (word_list[j])
(25)       End if
(26)    End for
(27)    Corpus.append (word_list)
(28) End for
(29) //Step 3: tokenization
(30) Tensorflow.load () //tensorflow is a machine learning framework
(31) Tokenizer = tensorflow.keras.pre-processing.text.Tokenizer() //Tokenizer is a class that converts text into sequences
(32) Tokenizer.fit_on_texts (corpus)
(33) Senquences = tokenizer.texts_to_sequences (corpus)
(34) Word_sequences = tensorflow.keras.pre-processing.sequence.pad_sequences (senquences, maxlen = l)
```

Algorithm 1: Data Preprocessing

Algorithm 1 for the pseudocode). Figure 3 shows an example of the preprocessing of a petition text.

(a) The cleaning process includes removing duplicate samples and null values, non-Chinese characters, and extra spaces. We first cleaned up these noises to prevent interference with the results. As shown in Figure 3(a), the horizontal tabs on both sides were removed.

(b) The word-splitting process involves splitting text into individual words. The Chinese texts require specialized operations for word splitting because of the absence of space. Jieba [37] is an open-source library for NLP that targets Chinese texts for word splitting. The library supports adding a custom user dictionary and removing stop-words. This study added 1495 words for the names of local streets and roads to the user dictionary. We also summarized three stop-word lists (the Chinese stop-word list, the stop-words list of Harbin Institute of Technology, and the stop-words list of the machine intelligence laboratory) with a total of 2490 stop-words. These words and symbols do not help the model to analyze and predict the category of the text but add computational complexity and system overhead. Figure 3(b) shows that after word splitting, such a sentence constitutes a row in the internal corpus.

(c) Tokenization represents a sentence as a sequence of words. First, all words in the corpus are obtained to form vocabulary V, and each word is assigned a unique integer number. Thus, a sentence is transformed into a sequence of word tokens. The tokenizer class in Keras facilitates this. Because the neural network can only receive inputs of the same length, we processed sentences of varying lengths into a sequence of standard lengths based on the length distribution of the petition text. Sequences less than the standard length were padded with zeros at the end of the sequences, and sequences longer than the standard length had extra words at the back-end cut-off.

Petition text:
/t/t Reflecting the morning broadcast of the Juquan Experimental Kindergarten was too loud, and asked for a solution instead of disturbing the residents. /t/t

(a) Clean:
Reflecting the morning broadcast of the Juquan Experimental Kindergarten was too loud, and asked for a solution instead of disturbing the residents.

(b) Word splitting:
Cut ~~Reflecting~~ Juquan Experimental Kindergarten morning broadcast ~~was too~~ loud ~~,~~ ~~and~~ asked for a solution instead of disturbing the residents ~~.~~
Remove stop-words
Juquan Experimental Kindergarten morning broadcast loud asked for solution instead of disturbing the residents

(c) Tokenization: [8, 7, 6, 3, 15, 10, 1, 9, 14, 376, 45, 79]
If standard length= 10: [8, 7, 6, 3, 15, 10, 1, 9, 14, 376]

Figure 3: Example of preprocessing operation.

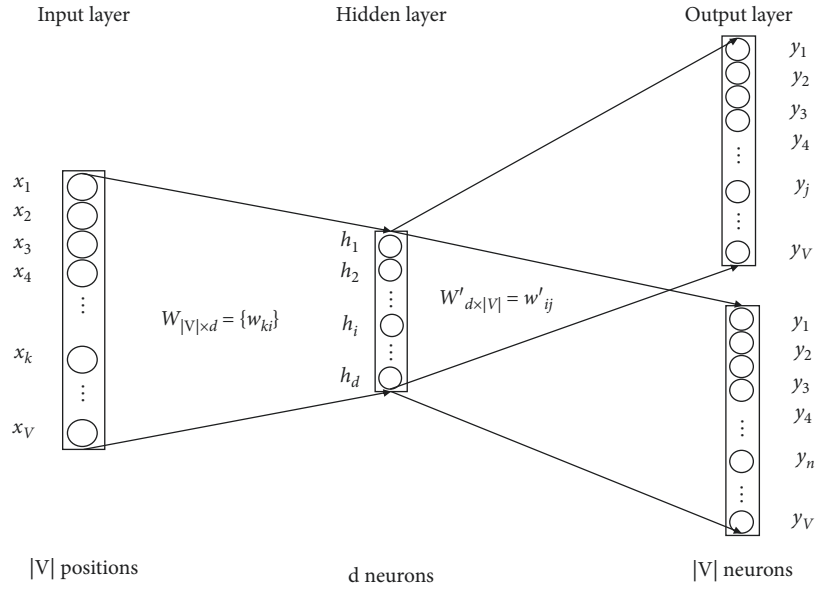

Figure 4: Word2vec embedding model training based on the skip-gram algorithm.

3.3. Word Embedding. To analyze text using machine learning algorithms, unstructured text must be converted into numerical form as input. Word embedding is a method of transforming words into vectors. In contrast to one-hot coding, word embedding allows text to be expressed low dimensionally. Word2vec is a mainstream statistical method for obtaining word vectors [38]. We employed word2vec based on the skip-gram algorithm to pretrain the embedding matrix. The skip-gram algorithm predicts the context words of the currently known word w_t in an internal corpus. The optimized objective function is as follows:

$$\mathscr{L} = \max \sum_{w \in c} \log p\left(\text{Context}\left(w\right) \mid w\right), \tag{1}$$

where $\text{Context}(w)$ is the context of the input word w. We defined a skip window parameter to represent the number of words selected from one side (left or right) of the current input word. Then, several contextual words from the whole window were selected as our output words; thus, there were multiple sets (input words and output words) of the training data. This neural network was trained using all training data. The model structure of the skip-gram is shown in Figure 4, which has three layers: the input, hidden, and output layers. The input layer was a one-hot vector of length $|V|$. Hidden layer was calculated as follows:

$$h = W^T \cdot X = V^T_{w_I}. \tag{2}$$

The weight $W \in R^{|V|\times d}$ from the input layer to the hidden layer is the embedding matrix, in which d represents the dimension of the word-embedding vector and $|V|$ is a constant-size vocabulary. $V^T_{w_i}$ is a distributed vector that represents the word w_I.

The output layer represents the probability that each word in dictionary V is an output word.

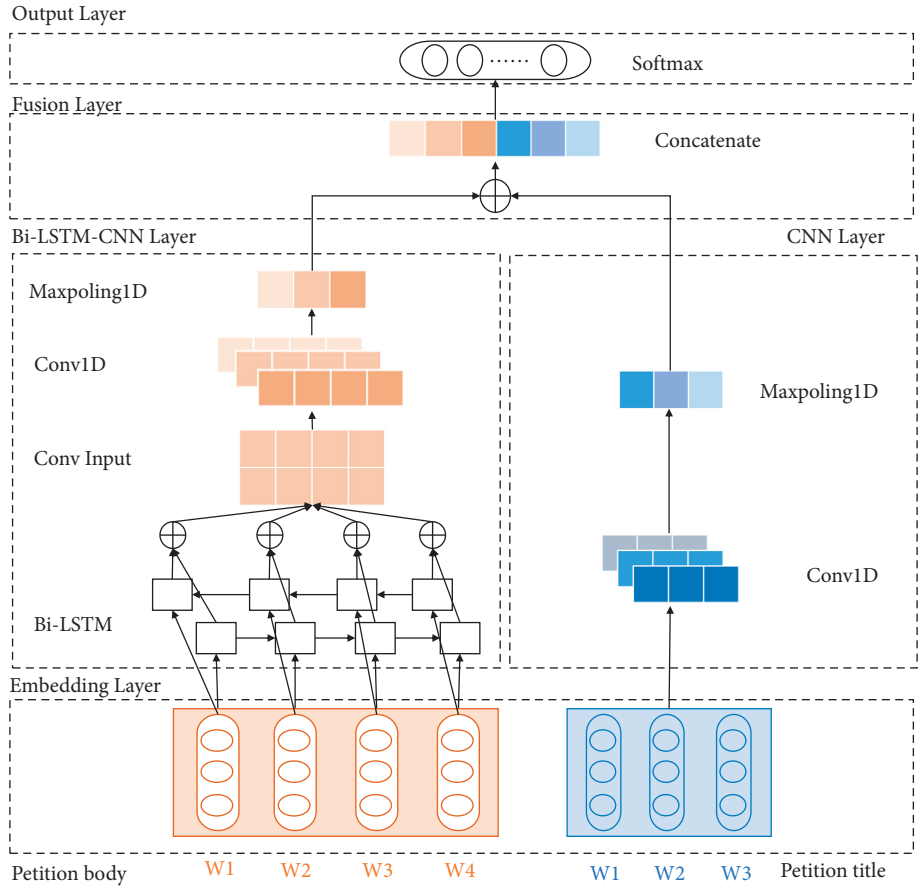

Figure 5: Architecture of the proposed hybrid deep learning classification method.

$$u_j = \sum_{i=1}^{d} W_{ij}'^{T} h_i,$$
$$P(w_j \mid w_I) = y_j = \frac{\exp(u_j)}{\sum_{k \in V} \exp(u_k)}. \tag{3}$$

3.4. Classification

3.4.1. Embedding Layer. This study used word2vec to pretrain an embedding matrix to initialize the weights of the embedding layers and fine-tune them. Each input sequence is mapped as a vector group through the embedding layer. Given a sequence consisting of l words: doc $= w_1, w_2, \ldots, w_l$, where each word w_i is associated with tokenized real-valued encoding, we looked up the embedding matrix W and transform w_i to e_i by matrix vector product:

$$e_i = W^T v_i. \tag{4}$$

In equation (4), v_i has an equal length to $|V|$ and the elements have values between 0 and 1. The sequence of words is represented as

$$\text{doc} = [e_1, e_2, \ldots, e_i, \ldots, e_l]^T. \tag{5}$$

Figure 5 shows that the petition title and petition body are represented as vector groups of different standard lengths, which are subsequently input into the CNN and Bi-LSTM-CNN layers, respectively.

3.4.2. CNN Layer. CNN was widely used in the early days of image processing, but it has only recently been used for text classification tasks and outperformed sequence-based methods. The convolution windows applied to text in the NLP domain are usually one dimensional. For the 1DCNN, the convolution layer uses a 1D cross-correlation operation, which slides the convolution window from top to bottom on the input text. The length of the 1D convolution window was consistent with the dimensions of the word vector and the width was n. The convolution operation applies the filter $F \in R^{d \times n}$ dots with n words in the text to obtain a new feature c_i:

$$c_i = f(F \cdot x_{i:n} + b). \tag{6}$$

Here, f is a nonlinear activation function such as relu or tanh, $x_{i:n}$ is the concatenation of n words: $x_{i:\,n} = x_i \oplus x_{i+1} \oplus x_{i+2} \oplus \ldots \oplus x_{i+n-1}$, and b is the bias term.

The convolution stride defines the distance between the positions of the convolution windows when they sweep through the adjacent feature map. In this study, the convolution step was set to 1, which means that the convolution

windows swept through the elements of the feature map one by one. We performed the same padding to maintain the output and input of the same size for feature extraction at the word level. Therefore, filter A generated one feature map $c = [c_1, c_2, \ldots, c_l]$. Finally, we performed a 1D maximum pooling operation on the feature map generated by the filter f_i to capture the most important features.

$$p_i = \max(c_1, c_2, \ldots, c_l). \tag{7}$$

Figure 6 shows an example of a filter that performs a one-dimensional convolution operation. The filter was multiplied by the word embeddings in the convolution window and summed to obtain a feature map containing locally dependent information. The maximum pooling operation selects the maximum value in the feature map to represent the local features captured by the filter. Filters are considerably important for CNN to automatically learn the features required to understand the text. Therefore, the size and number of filters are important parameters affecting the performance of CNN.

3.4.3. Bi-LSTM-CNN Layer. The unidirectional LSTM network accesses only forward information; however, reverse information is also needed in most cases to fully understand the context. Bi-LSTM effectively uses past and future contextual information to learn semantic information. LSTM is a variant of RNN that introduces a gating mechanism to control the accumulation speed of information, including the input gate i_t, forget the door f_t, and output gate o_t. The LSTM network introduces a new internal state c_t to simultaneously transmit recurrent information and output information to the external state of the hidden state h_t at the same time. The formula used is as follows:

$$\begin{aligned} i_t &= \sigma(W_i x_t + U_i h_{t-1} + b_i), \\ f_t &= \sigma\left(W_f x_t + U_f h_{t-1} + b_f\right), \\ o_t &= \sigma(W_o x_t + U_o h_{t-1} + b_o, \\ \widetilde{c_t} &= tanh(W_c x_t + U_c h_{t-1} + b_c), \\ c_t &= f_t \odot c_{t-1} + i_t \odot \widetilde{c_t}. \end{aligned} \tag{8}$$

Here, σ is the sigmoid function. x_t is the input of the current time t, h_{t-1} is the hidden state of the last moment, $\odot$ is the product of vector elements, c_{t-1} is the memory unit of the last moment, and $\widetilde{c_t}$ is a candidate state obtained by a nonlinear function. It selectively adds new information and forgets previously accumulated information, which can effectively solve the problems of gradient disappearance and gradient explosion of a simple recurrent neural network.

Assuming that the first layer is in the order of sentences and the second layer is in the reverse order of sentences, the hidden state at time t is defined as $h_t^{(1)}$ and $h_t^{(2)}$, then

$$\begin{aligned} h_t^{(1)} &= f\left(U^{(1)} h_{t-1}^{(1)} + W^{(1)} x_t + b^{(1)}\right), \\ h_t^{(2)} &= f\left(U^{(2)} h_{t-1}^{(2)} + W^{(2)} x_t + b^{(2)}\right), \\ h_t &= h_t^{(1)} \oplus h_t^{(2)}, \end{aligned} \tag{9}$$

where $\oplus$ is a vector concatenate operation.

When using LSTM for text classification tasks, the hidden-layer state of the last time step is typically used as the penultimate layer. However, this study extracted local features using a 1DCNN based on the contextual features of long texts. Therefore, the hidden state of each time step was returned, and h_t was the concatenation of the hidden states of the forward and backward units at time step t. A text sequence of length l returns l hidden states after context features are extracted by Bi-LSTM, and these returned hidden states can be regarded as a sequence of words of length l with embedding dimension 2 h. Next, we performed the convolution and maximum pooling operations described in Subsection 3.4.2. The 1DCNN was performed on returned hidden states and then a maximum pooling operation to capture the prominent features of the petitioning body.

3.4.4. Fusion Layer. This study used different neural network structures to extract features according to the length characteristics of various petition parts. The Bi-LSTM-CNN layer uses the petition body as input, extracts contextual information from the input using forward and backward LSTMs, and returns the hidden state of all time steps into a 1DCNN for convolution and maximum pooling. Meanwhile, the CNN layer uses the petition title as input, performs a convolution operation on the input, and calculates the maximum value of each activation mapping using 1D maximum pooling. To obtain the advantages of both shallow neural networks, the output of the Bi-LSTM-CNN layer, noted O_b, and the output of the CNN layer, noted O_t, are concatenated to create a unified representation Y. Finally, the resulting unified representation is transported to the output layer.

$$Y = O_b \oplus O_t. \tag{10}$$

3.4.5. Output Layer. The proposed end-to-end framework for automatic classification supports multiple petition classifications. The fully connected layer integrates the category differentiation information in Y and then outputs y_i. The calculation formula is as follows:

$$y_i = W_{(y_i)}^T Y, \tag{11}$$

where $W_{(y_i)}^T$ is the learnable weight connected to the neuron y_i. The softmax activation function was used to generate a probability p_i for each label. Finally, the label with the highest probability among the k labels was selected as the final reply department of petition P.

$$\begin{aligned} p_i &= \frac{\exp(y_i)}{\sum_{j \in k} \exp\left(y_j\right)}, \\ y &= \max(p_1, p_2, \ldots, p_k). \end{aligned} \tag{12}$$

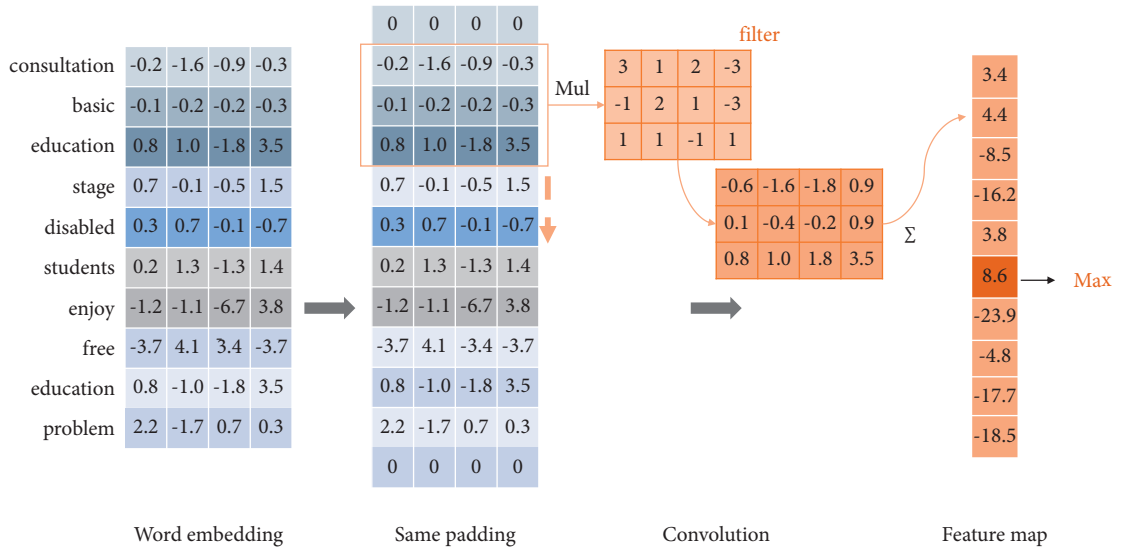

Figure 6: An example of using a filter to extract features.

4. Numerical Experiments

4.1. E-Petition Data Set. In 2016, Shanghai integrated various channels to establish a comprehensive e-petitioning platform. This study conducted an experiment using open petition data collected from May 2016 to September 2021. We chose departments that received over 1000 e-petitions as labels for the classifier, and 11 departments were selected. After removing duplicate and missing values, 28085 records were collected as the data set for this study, and Table 3 shows the categories and data volume statistics.

4.2. Data Preparation. The number of words in the corpus of the fully structured e-petition data was 110,645. Figure 7 shows the text length distributions of the title and body. The title is truncated at 10, and the text is truncated at 500, but with a relatively long right tail. Excessive length increases the computational overhead; therefore, we used 10 and 500 as the standard length of the title sequence and body sequence, respectively.

For the preprocessed petition data set, 20% of the shuffling data set was selected as the test set and 80% as the training set. The validation set comprised the last 10% of the training set. The proportions of the categories in the three data sets remained similar. Table 4 summarizes the statistics and splitting of the data sets.

4.3. Classification and Comparison

4.3.1. Parameter Setting. Some basic parameter settings are summarized in Table 5, which serve as the default settings for the proposed classification model and work well in many cases. Owing to the limited amount of sample data, which contains noisy data, it is easy to obtain high training accuracy on an unknown data set. Early stopping is used to prevent overfitting. During the gradient descent training process, the parameters that converged on the training set were not necessarily the best on the test set. This study used validation loss as the monitor. In each epoch, we estimated the newly obtained model for the validation set and computed the validation loss. If the validation loss did not decrease, then the iteration was stopped. To solve the problem of class imbalance, class weight was considered to weigh the loss function during training to make the model more attentive to the samples of underrepresented classes. In this study, the class weight was set to "balanced" and the weight of category i was calculated as follows:

$$w_i = \frac{N}{\text{labels} * n_i}, \tag{13}$$

where N is the total number of samples, labels are the total number of categories, and n_i is the number of samples in category i.

In addition to the aforementioned basic settings, some important parameters were sought to improve the performance of the model (see Table 6). The introduction of a dropout before the output layer can alleviate this overfitting problem. The problem of gradient explosion appears in model training when the activation function in LSTM is set as relu; therefore, tanh was set as the default. The activation function of the CNN layer was optimized for relu and tanh, as recommended by [15]. To select the optimal parameter value to achieve high accuracy, we trained the model on the training set and determined the optimal parameter settings based on the classification performance of the validation set.

4.3.2. Evaluation Measures. The evaluation measures include accuracy, precision, recall, and $F1$ score; category i to be observed was considered positive, and the rest of the categories were considered negative, and a confusion matrix similar to binary classification was obtained. The evaluation metrics were calculated as follows:

Table 3: Classes and volume of the petition data set.

Reply departments	Volume	Label	Proportion (%)
Municipal Finance Bureau	1174	0	4.18
Public Security Bureau	3033	1	10.80
Municipal Education Commission	4765	2	16.97
Municipal Human Resources and Social Security Bureau	1707	3	6.08
Municipal Administration for Market Regulation	5597	4	19.93
Municipal Health Commission	1437	5	5.12
Municipal Commission of Housing and Urban-Rural Development	3992	6	14.21
Municipal Transportation Commission	2446	7	8.71
Landscaping & City Appearance Administrative Bureau	822	8	2.93
Urban Management and Law Enforcement Bureau	1252	9	4.46
Planning and Natural Resources Bureau	1860	10	6.62

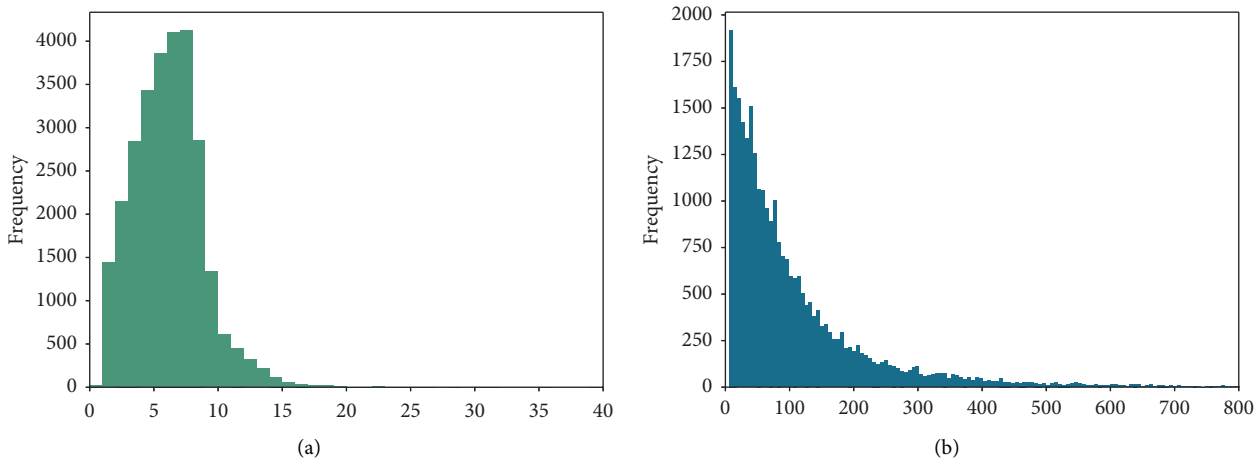

Figure 7: Length distribution of word sequences. (a) Length of title; (b) length of body.

Table 4: Statistics and splitting of e-petition data set.

Data set	Training set	Validation set	Test set	Vocab	Title length	Body length
28085	20221	2247	5617	110645	10	500

Table 5: Basic parameter settings of the classification model.

Parameter	Value
Loss function	Categorical cross-entropy
Metrics	Categorical accuracy
Optimizer	Adam
Early stopping	monitor = 'val-loss', patience = 1
Class weight	Balanced

Table 6: Key parameters of the classification model to be optimized.

Parameter	Explanation	Scope
Batch size (b)	Number of samples per gradient update	$16 \le b \le 64$, $b \subset Z$
Learning rate (lr)	Hyperparameters in updating weights during gradient descent	$0.0001 \le b \le 0.01$
Embedding size (e)	The dimensionality of the word vector	$100 \le e \le 600$, $e \subset Z$
Hidden size (h)	Number of neurons in the hidden layer	$32 \le h \le 256$, $h \subset Z$
Kernel size (c_1)	Length of convolution window in Bi-LSTM-CNN layer	$2 \le c_1 \le 10$, $c_1 \subset Z$
Kernel size (c_2)	Length of convolution window in CNN layer	$1 \le c_1 \le 6$, $c_1 \subset Z$
Filter size (f_1)	Number filters in Bi-LSTM-CNN layer	$10 \le f_1 \le 400$, $f_1 \subset Z$
Filter size (f_2)	Number filters in CNN layer	$5 \le f_2 \le 200$, $f_2 \subset Z$
Dropout (d)	A simple way to prevent neural networks from overfitting	$0 \le d \le 0.8$
Activation function (a)	A function for introducing nonlinear factors into a neural network	Relu or tanh

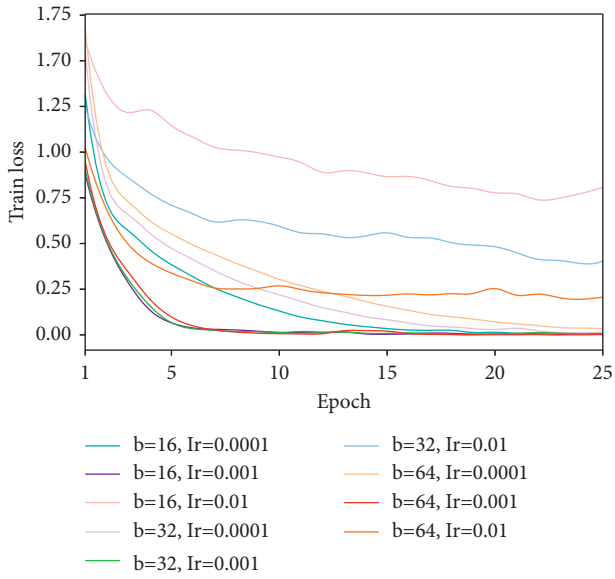

FIGURE 8: Training loss reduction curves for different combinations of batch size and learning rate.

$$\begin{aligned} \text{Accuracy} &= \frac{\text{TP} + \text{TN}}{N}, \\ P_i &= \frac{\text{TP}}{\text{TP} + \text{FP}}, \\ R_i &= \frac{\text{TP}}{\text{TP} + \text{FN}}, \\ F1_i &= \frac{2P_i \times R_i}{P_i + R_i}. \end{aligned} \tag{14}$$

Considering the uneven distribution of the samples, we introduced weights to each evaluation index to evaluate the overall classification effect of the samples. Here, the weights were determined by the sample proportion, p.

$$\begin{aligned} \text{Weighted} - P &= \sum_{i=0}^{10} p_i P_i, \\ \text{Weighted} - R &= \sum_{i=0}^{10} p_i R_i, \\ \text{Weighted} - F1 &= \sum_{i=0}^{10} p_i F1_i. \end{aligned} \tag{15}$$

4.3.3. Parameter Optimization. To determine the optimal settings of many hyperparameters in the model step-by-step, we randomly initialized a set of parameters and selected the optimal choice in the current state as the setting of the parameters. Figure 8 shows the decline of the model in the training loss over 25 epochs with nine sets of batch sizes and learning rates. The results show that the model reached the lowest loss fastest at a learning rate of 0.001, with three batch size settings. When the batch size was 64, the training time was shorter than that for the other two batch size settings. Considering the training time of the model, the final choice was a learning rate of 0.001 and a batch size of 64.

Next, the remaining parameters were optimized sequentially. Figure 9(a) shows the classification accuracy of the configuration of the embedding size on the validation set. The model had the highest validation accuracy when the embedding size was set to 400. Figure 9(b) shows that the classification results were slightly better when the hidden size was set to 128. For the kernel size, the convolution size of the body had a significant impact on the classification performance of the model, fluctuating by 5%. We selected $c1 = 5$ and $c2 = 3$ as the optimal parameter settings. Figures 10(a) and 10(b) show that when the number of filters was below 100, the accuracy was positively correlated with the number of filters, and the optimal number of filters was set to 100. Figure 10(c) shows that the model performed best when dropout = 0.25. Figure 10(d) shows that the two activation functions have minimal impact on model performance, and the activation function set as tanh is slightly better.

4.3.4. Classification Result. We controlled the three embedded modes of the proposed structure in the embedding layer and obtained the classification results for the test set (see Table 7). Word2vec pretrained word embedding with fine-tuning in the embedding layer had the highest classification accuracy, weighted precision, weighted recall, and weighted-$F1$ score, and the randomly initialized word embedding approach had the poorest classification performance. In addition, pretrained word embeddings, either static or fine-tuned, can reduce the training time of the model.

4.3.5. Comparisons and Discussion. This study compared baseline models with the proposed model to verify its classification performance. The basic settings of the baseline models were consistent with those of the proposed model (see Table 5), and the other settings are listed in Table 8. The input of baseline models represented as "body + title", which respectively indicates the petition body and petition title.

The classification results of the proposed and baseline models for the test set are summarized in Table 9. The baseline models provided reasonable results, and the proposed method was more effective. First, in the case of using the body as the input text for the model, hybrid models (M7, M10) outperformed the single structure models (M1 and M4) in each evaluation. The results showed that Bi-LSTM outperforms Text-CNN, which demonstrates that extracting contextual features is more effective for classification tasks for long petition bodies. In addition, the hybrid approach of Bi-LSTM for extracting contextual features, followed by CNN for extracting local features, showed better results. Second, introducing title information can improve the classification effect (M2, M3, M5, M6, M8, M9, M11, and proposed model). Compared to the Bi-LSTM-CNN, the proposed model improves 1.26%, 1.67%, and 1.58% in weighted precision, weighted recall, and weighted $F1$ score, respectively. Text-CNN, which performed the worst in text classification using the body of the petition, improved the recall by 3.22% after adding the title. Third, among the models with titles, the evaluations of models with the CNN

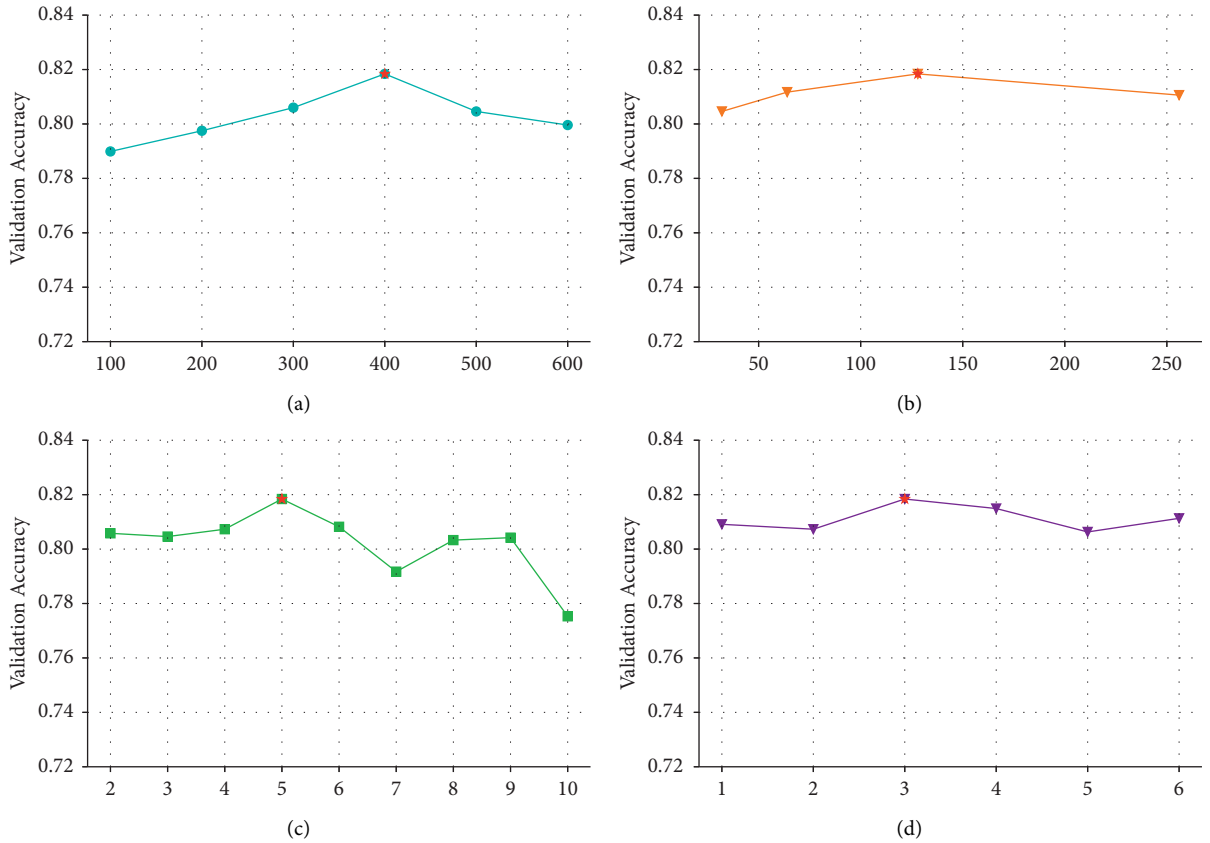

Figure 9: Parameter optimization of embedding size (e), hidden size (h), and convolutional kernel size ($c1$, $c2$). (a) Embedded size (e); (b) hidden size (h); (c) kernel size ($c1$); (d) kernel size $c2$.

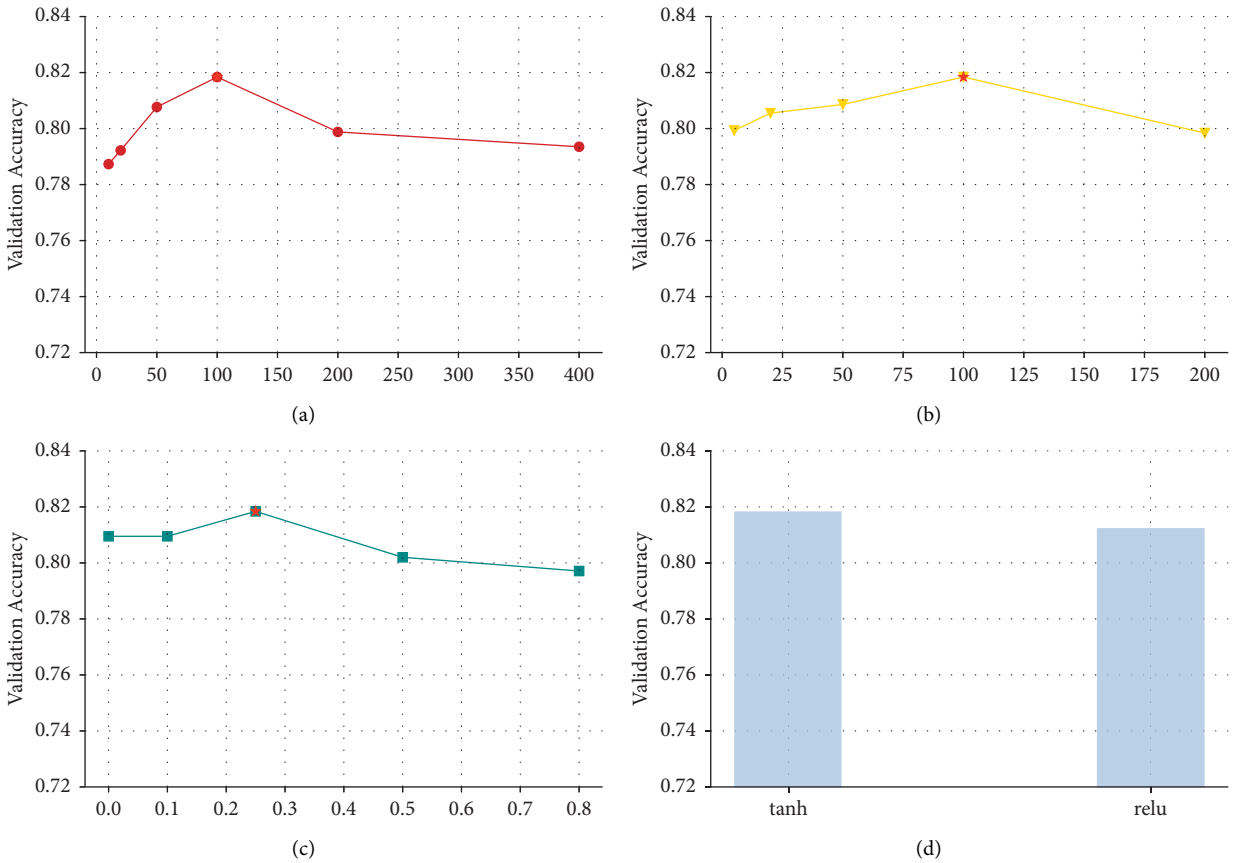

Figure 10: Parameter optimization of filters ($f1$, $f2$), dropout d, and activation function a. (a) Filters ($f1$); (b) filters ($f2$); (c) dropout (d); (d) activation function (a).

Table 7: Classification results of three embed modes of the embedding layer on the test set.

Embed mode	Weighted-P	Weighted-R	Weighted-$F1$	Training time (second)
Random	0.8137	0.8104	0.8103	1249
Static	0.8226	0.8214	0.8212	430
Fine-tuning	0.8288	0.8262	0.8267	1100

Table 8: Configurations of baseline models.

Baseline models	Parameter setting
Text-CNN (M1)	$e = 400, c_1 = 3, 4, 5$, filters = 100, input = petition body
Text-CNN + LSTM (M2)	$e = 400, c_1 = 3, 4, 5, h = 128$, input = petition body and title
Text-CNN + CNN (M3)	$e = 400, c_1 = 3, 4, 5, c_2 = 3$, filters = 100, input = petition body and title
Bi-LSTM (M4)	$e = 400, h = 128$, input = petition body
Bi-LSTM + LSTM (M5)	$e = 400, h_1 = 128, h_2 = 128$, input = petition body and title
Bi-LSTM + CNN (M6)	$e = 400, h = 128, c_2 = 3$, input = petition body and title
CNN - Bi-LSTM (M7)	$e = 400, c_1 = 5, h = 128, c_2 = 3$, input = petition body
CNN - Bi-LSTM + LSTM (M8)	$e = 400, c_1 = 5, h_1 = 128, h_2 = 128$, input = petition body and title
CNN - Bi-LSTM + CNN (M9)	$e = 400, c_1 = 5, h = 128, c_2 = 3$, input = petition body and title
Bi-LSTM-CNN (M10)	$e = 400, c_1 = 5, h = 128$, input = petition body
Bi-LSTM-CNN + LSTM (M11)	$e = 400, c_1 = 5, h_1 = 128, h_2 = 128$, input = petition body and title

Table 9: Evaluation of the proposed model and baseline models in the test set.

Models	Weighted-P	Weighted-R	Weighted-$F1$
M1	0.7856	0.7668	0.7685
M2	0.8006	0.7899	0.7920
M3	0.8026	0.7990	0.7988
M4	0.7892	0.7839	0.7836
M5	0.8044	0.7953	0.7971
M6	0.8124	0.8109	0.8104
M7	0.7883	0.7821	0.7825
M8	0.7986	0.7942	0.7952
M9	0.7941	0.7887	0.7894
M10	0.8162	0.8095	0.8109
M11	0.8169	0.8124	0.8131
Proposed model	0.8288	0.8262	0.8267

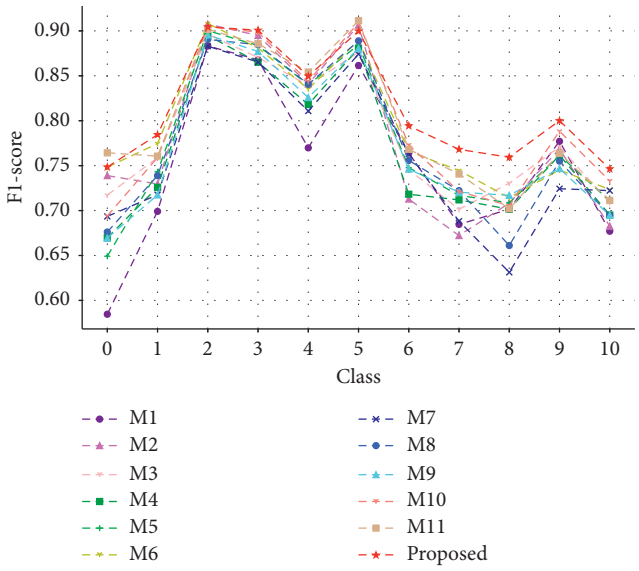

Figure 11: $F1$ score of proposed model and baseline models in each class on the test set.

structure (M3, M6, proposed model) are better than those of models with the LSTM structure (M2, M5, M11). Compared with the baseline models, the proposed model achieved the best classification results in the petition text classification task.

We summarized the $F1$ score for each class (see Figure 11) to investigate the classification performance of each model in the unbalanced data set. $F1$ score balances the precision and recall of the classification model. The results show that the proposed model had a high $F1$ score of over 0.74 in each class. For Classes 2, 3, and 5, where the classification results were generally good, the $F1$ scores of the proposed model were close to 0.9. The $F1$ scores of the proposed model significantly improved for classes 0, 7, 8, and 10. Our model provides more comprehensive classification results for biased samples.

Considering the independence of each government department, evaluating the classification effects of all departments is beneficial for discovering the causes of differences in classification effects among departments. We calculated the confusion matrix for the classification results. Each matrix element was divided by the sum of each column, i.e., the misclassification rate of each category (see Figure 12). Although the proposed model significantly improves the misclassification rate of class 6 compared to the other baseline models, all models tend to misclassify classes 6, 7, and 10. Analyzing the classification results of the proposed model, there were 41 petitions for "tenement" and "facility management" in class 7 that were mistakenly classified as class 6 (8.12%) and 57 petitions for housing relocation in class 10 that were mistakenly classified as class 6 (15.45%). We investigated the duties of the department of Class 6, which included coordinating various departments to manage land use and supervising construction and housing issues. One of the duties of the department of Class 10 is planning the construction of engineering projects, and the department of Class 7's duties are related to parking lot

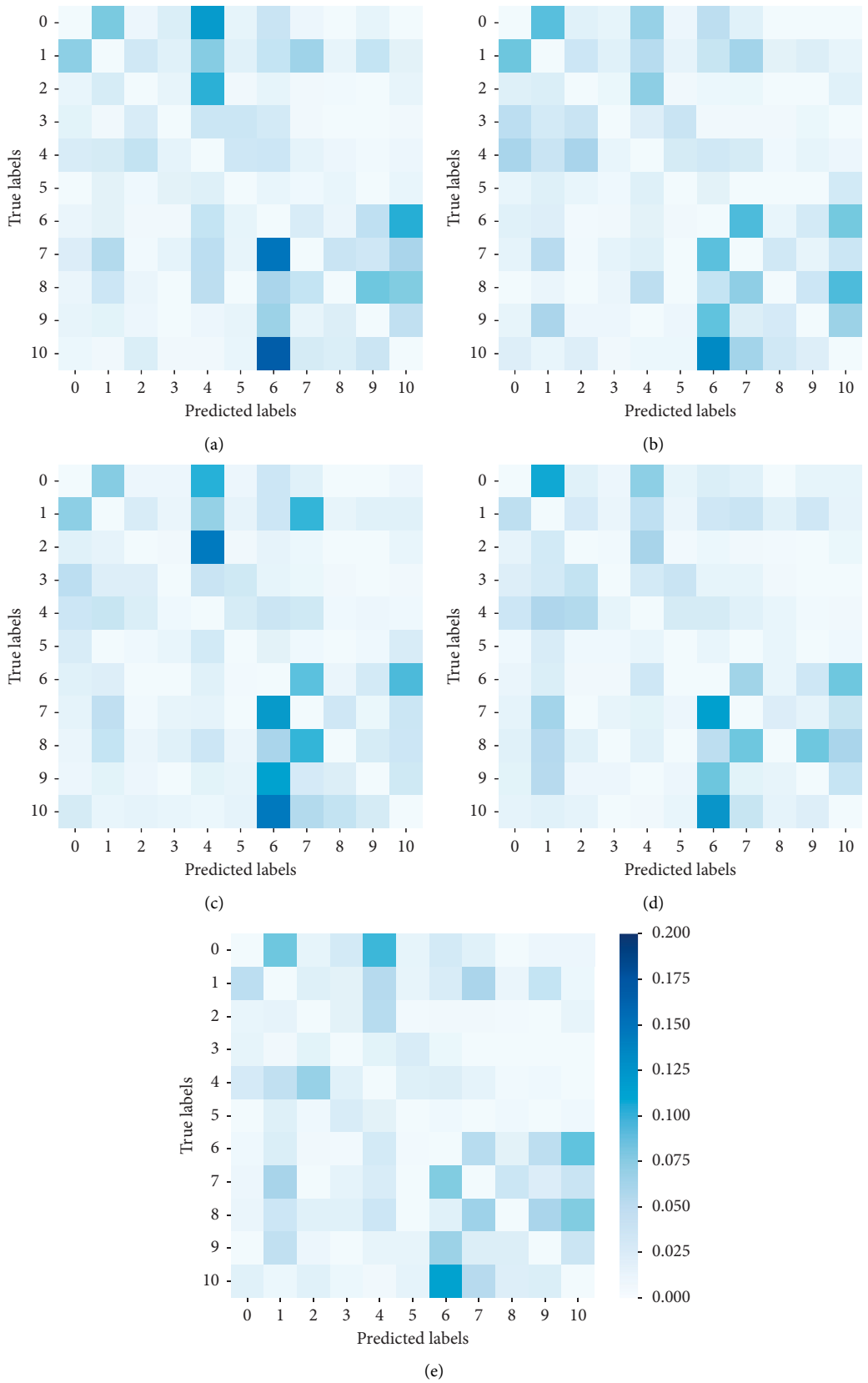

Figure 12: Misclassification of the proposed and baseline models on the test set: (a) Text-CNN; (b) Bi-LSTM; (c) CNN-Bi-LSTM; (d) Bi-LSTM-CNN; (e) Proposed model.

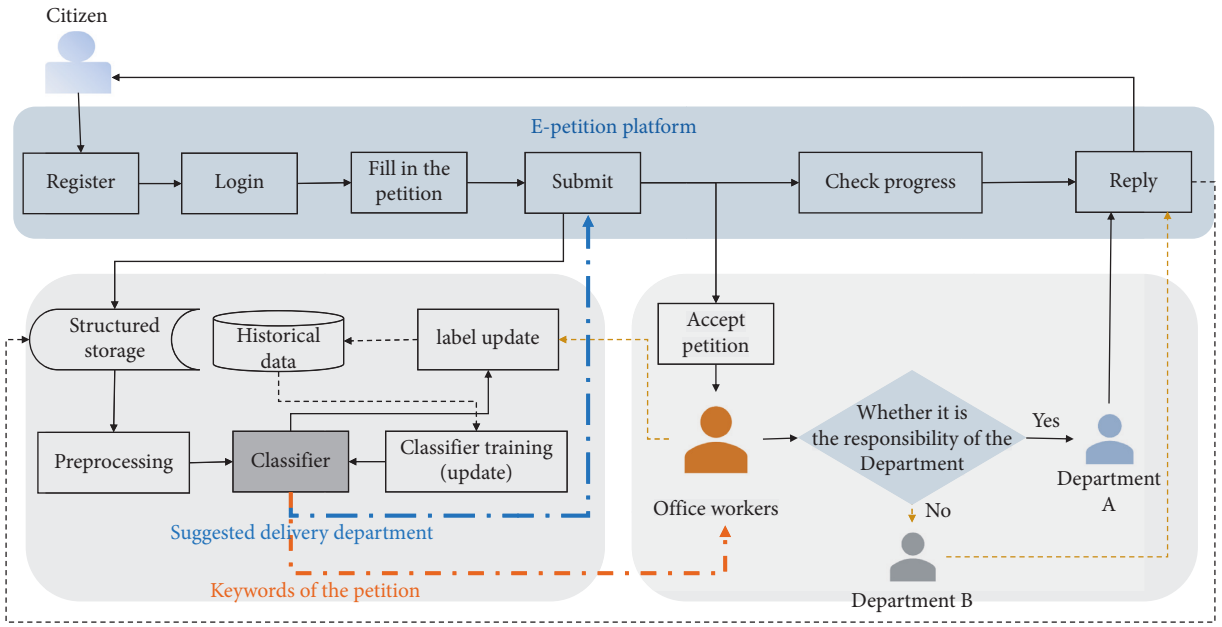

Figure 13: Petition automated forwarding decision support system.

operations in urban areas. Their duties overlap when dealing with housing issues. Clear and well-defined responsibilities are important for the effective automatic classification of petitions.

4.4. Implications and Applications. Efficient interaction between citizens and the government can help the government analyze public opinion, understand and solve difficulties in people's lives, and make favorable decisions. Assessing the "e-petition platform" in China, citizens are required to choose a responsible department when submitting a petition, and office workers receive the petition, sort it, and send it to the responsible department according to the demand. Because citizens do not have a clear understanding of the department's responsibility, they sometimes choose the wrong responsible department. When petition office workers sort many petitions, some petitions have unknown needs and cannot be quickly processed.

Using the automatic text classification model proposed in this study and the current application requirements, we designed a text-based petition decision support system to help citizens in selecting a responsible department and to help office staff improve the sorting speed of petitions with unclear demands. As shown in Figure 13, on the one hand, after citizens register and log in to the e-petition platform, they submit a petition with a title and body. The decision support system uses a classifier trained from historical data to predict the responsible department, provides the predicted result to the citizen for reference, and then stores label A. Moreover, the submitted petition is received and registered by the office staff, which provides reference keywords to help the staff improve the sorting speed and handle petitions with unclear needs. If the staff judges that the petition does not fall under the authority of Department A, the petition is forwarded to Department B, and the label is updated. Meanwhile, the data are added to the training set to update the classifier. Before autonomous classification, we first store the data in a structured manner and store the labels and replies, which accumulate data resources for further analysis of public opinion, prediction of hot topics, and evaluation of department performance. This will become an integral part of the e-government decision support system. The system will also continuously update the data to improve the accuracy of autonomous classification, which remarkably improves the efficiency of petition administrators.

In addition to the application framework, we also provide some useful application suggestions based on the research results and findings. First, the data of each department is unbalanced. We should strengthen the ability to identify categories with small sample sizes, balance the proportion of samples in each category, and be alert to the extreme prediction of the classifier as a department with a large sample size. Second, we set a classification threshold for a single petition. Departments with more overlapping responsibilities had similar predicted probabilities. When the difference between the predicted probabilities of multiple classes is small, we set a threshold for delivery to multiple departments.

5. Conclusion

Communication and interaction between citizens and government are important for social governance. This study researched questions for petition autonomous delivery and decision support by reviewing the gap between existing research and application requirements. We proposed a hybrid model that integrates the advantages of both neural networks. The petitions were unstructured. Therefore, we preprocessed the text and mapped the words into low-dimensional vectors. Then, we input the word

embedding of the title and body into the hybrid deep learning model, extracting the contextual and local features to realize the autonomous classification of the petition-responsible department. Numerical experimental results show that compared with the baseline models, the overall $F1$ score of the proposed method is 0.8267, which is 5.82% higher than that of Text-CNN, 4.31% higher than that of Bi-LSTM, and 1.58% higher than that of Bi-LSTM-CNN.

Further, we explored the superiority of the model in various classes and found that the hybrid model outperformed the simple model. The proposed model incorporates the title and body of the text well and had better classification performance. Finally, we designed a decision-making support system that can store data and update classifiers, thereby significantly promoting the automation and efficiency of the petition workflow. We also provided useful suggestions for applications in real-world scenarios. In conclusion, our research provides a significant practical reference.

Furthermore, we must note that departments receiving a few e-petitions are not included in this study and that overlapping responsibilities among departments may be biased in practice. Therefore, further analysis should focus on small samples and the problem of e-petitions corresponding to multiple departments.

Data Availability

Data can be obtained via the request of the corresponding author.

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 (grant nos. 52131101, 51939001, and U1813203), the Liaoning Revitalization Talents Program (grant nos. XLYC1807046, and XLYC1908018), the Natural Science Foundation of Liaoning Province (grant no. 2020-HYLH-26), and the Science and Technology Fund for Distinguished Young Scholars of Dalian (grant no. 2021RJ08). Thanks to Jinsong Jia, Jinling Zhou, Yuchuan Hu, Weizheng Zhao, and Licheng Zhao for the experiment assistance and technique discussion. We acknowledge editors Afaq Ahmad and Toqeer Mahmood and reviewers for the valuable advice and publishing support.

References

[1] J. Jiang, T. Meng, and Q. Zhang, "From Internet to social safety net: the policy consequences of online participation in China," *Governance*, vol. 32, no. 3, pp. 531–546, 2019.

[2] C. Leston-Bandeira, "Parliamentary petitions and public engagement: an empirical analysis of the role of e-petitions," *Policy & Politics*, vol. 47, no. 3, pp. 415–436, 2019.

[3] C. Puschmann, M. T. Bastos, and J. H. Schmidt, "Birds of a feather petition together? Characterizing e-petitioning through the lens of platform data," *Information, Communication & Society*, vol. 20, no. 2, pp. 203–220, 2016.

[4] C. C. Aggarwal and C. X. Zhai, "A survey of text classification algorithms," in *Mining Text Data*, C. Aggarwal and C. Zhai, Eds., Springer, Boston, MA, 2012.

[5] J. Hartmann, J. Huppertz, C. Schamp, and M. Heitmann, "Comparing automated text classification methods," *International Journal of Research in Marketing*, vol. 36, no. 1, pp. 20–38, 2019.

[6] M. A. Ibrahim, M. U. Ghani Khan, F. Mehmood, M. N. Asim, and W. Mahmood, "GHS-NET a generic hybridized shallow neural network for multi-label biomedical text classification," *Journal of Biomedical Informatics*, vol. 116, Article ID 103699, 2021.

[7] P. Barberá, A. E. Boydstun, S. Linn, R. Nagler, and J. Nagler, "Automated text classification of news articles: a practical guide," *Political Analysis*, vol. 29, no. 1, pp. 19–42, 2021.

[8] S. Clark, N. Lomax, and M. A. Morris, "Classification of Westminster Parliamentary constituencies using e-petition data," *EPJ Data Science*, vol. 6, no. 1, p. 16, 2017.

[9] J. Bughin, E. Hazan, S. Ramaswamy et al., *Artificial Intelligence: The Next Digital Frontier?* McKinsey Global Institute, McKinsey & Company, Atlanta, 2017.

[10] T. Joachims, "Text categorization with support vector machines: learning with many relevant features," in *Lecture Notes in Computer Science*, C. Nédellec and C. Rouveirol, Eds., vol. 1398, Berlin, Heidelberg, Springer, 1998.

[11] V. D'Orazio, S. T. Landis, G. Schrodt, and P. Schrodt, "Separating the wheat from the chaff: applications of automated document classification using support vector machines," *Political Analysis*, vol. 22, no. 2, pp. 224–242, 2014.

[12] S. Tan, "An effective refinement strategy for KNN text classifier," *Expert Systems with Applications*, vol. 30, no. 2, pp. 290–298, 2006.

[13] Z. Yang, X. Nie, W. Xu, and J. Guo, "An approach to spam detection by naive Bayes ensemble based on decision induction," in *Proceedings of the 6th International Conference on Intelligent Systems Design and Applications*, pp. 861–866, Jian, China, October 2006.

[14] G. E. Hinton, R. R. Salakhutdinov, and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[15] Y. Kim, "Convolutional neural networks for sentence classification," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pp. 1746–1751, Doha Qatar, Aug 2014.

[16] J. Cheng, P. Li, Z. Ding, S. Zhang, and H. Wang, "Sentiment classification of Chinese microblogging texts with global RNN," in *Proceedings of the 2016 IEEE First International Conference on Data Science in Cyberspace*, pp. 653–657, Changsha, China, June 2016.

[17] C. Masterson and M. Masterson, "Using word order in political text classification with long short-term memory models," *Political Analysis*, vol. 28, no. 3, pp. 395–411, 2020.

[18] H. Ye, B. Cao, Z. Peng, T. Chen, Y. Liu, and J. Liu, "Web services classification based on wide & Bi-LSTM model," *IEEE Access*, vol. 7, pp. 43697–43706, 2019.

[19] J. L. Cui, S. Qiu, M. Y. Jiang, Z. L. Pei, and Y. N. Lu, "Text classification based on ReLU activation function of SAE algorithm," *Advances in Neural Networks - ISNN 2017*, vol. 10261, pp. 44–50, 2017.

[20] M. Y. Jiang, Y. C. Liang, X. Y. Feng et al., "Text classification based on deep belief network and softmax regression," *Neural Computing & Applications*, vol. 29, no. 1, pp. 61–70, 2018.

[21] D. Larabi-Marie-Sainte and S. Larabi-Marie-Sainte, "Arabic text classification using convolutional neural network and genetic algorithms," *IEEE Access*, vol. 9, pp. 91670–91685, 2021.

[22] A. Bilbao-Jayo and A. Almeida, "Automatic political discourse analysis with multi-scale convolutional neural networks and contextual data," *International Journal of Distributed Sensor Networks*, vol. 14, no. 11, Article ID 155014771881182, 2018.

[23] W. Huang and H. Huang, "A gating context-aware text classification model with BERT and graph convolutional networks," *Journal of Intelligent and Fuzzy Systems*, vol. 40, no. 3, pp. 4331–4343, 2021.

[24] B. Jang, M. Kim, G. Harerimana, S. u. Kang, and J. W. Kim, "Bi-LSTM model to increase accuracy in text classification: combining Word2vec CNN and attention mechanism," *Applied Sciences*, vol. 10, no. 17, p. 5841, 2020.

[25] P. Zhou and N. El-Gohary, "Domain-specific hierarchical text classification for supporting automated environmental compliance checking," *Journal of Computing in Civil Engineering*, vol. 30, no. 4, 2016.

[26] P. Blunsom, E. Grefenstette, and N. Kalchbrenner, "A convolutional neural network for modelling sentences," pp. 655–665, 2014, https://arxiv.org/abs/1404.2188.

[27] M. M. Mostafa, "More than words: social networks' text mining for consumer brand sentiments," *Expert Systems with Applications*, vol. 40, no. 10, pp. 4241–4251, 2013.

[28] S. Lu and Y. Lu, "A modularized architecture of multi-branch convolutional neural network for image captioning," *Electronics*, vol. 8, no. 12, p. 1417, 2019.

[29] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) fromscratch," *Journal of Machine Learning Research*, vol. 12, pp. 2493–2537, 2011.

[30] A. C. Metlapalli, T. Muthusamy, and B. P. Battula, "Classification of social media text spam using VAE-CNN and LSTM model," *Ingénierie des Systèmes d'Information*, vol. 25, no. 6, pp. 747–753, 2020.

[31] C. Li, G. Zhan, and Z. Li, "News text classification based on improved Bi-LSTM-CNN," in *Proceedings of the 9th International Conference on Information Technology in Medicine and Education*, pp. 890–893, Hangzhou, China, October 2018.

[32] X. Han, J. Wang, M. Zhang, and X. Wang, "Using social media to mine and analyze public opinion related to COVID-19 in China," *International Journal of Environmental Research and Public Health*, vol. 17, no. 8, p. 2788, 2020.

[33] H. Xuan Huynh, V. T. Nguyen, N. Duong-Trung, V.-H. Phan, and C. T. Phan, "Distributed framework for automating opinion discretization from text corpora on facebook," *IEEE Access*, vol. 7, pp. 78675–78684, 2019.

[34] L. Bencke, C. Cechinel, and R. Munoz, "Automated classification of social network messages into smart cities dimensions," *Future Generation Computer Systems*, vol. 109, pp. 218–237, 2020.

[35] E. D'Andrea, P. Ducange, A. Bechini, A. Renda, and F. Marcelloni, "Monitoring the public opinion about the vaccination topic from tweets analysis," *Expert Systems with Applications*, vol. 116, pp. 209–226, 2019.

[36] Y. Zhang, Z. Yu, C. Mao, Y. X. Huang, and S. X. Gao, "Correlation analysis of law-related news combining bidirectional attention flow of news title and body," *Journal of Intelligent and Fuzzy Systems*, vol. 40, no. 3, pp. 5623–5635, 2021.

[37] Jieba, "Chinese word segmentation tool," 2012, https://github.com/fxsjy/jieba.

[38] T. Mikolov, K. Chen, G. S. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," in *Proceedings of the 2013 International Conference on Learning Representations*, Scottsdale, Arizona, U.S, 2013, https://arxiv.org/abs/1301.3781.