Wireless Communications and Mobile Computing

Generative Adversarial Networks for Multi-Modal Multimedia Computing

Lead Guest Editor: Philippe Fournier-Viger
Guest Editors: Jerry Lin, Rage Uday Kiran, and Yulin Wang

Generative Adversarial Networks for Multi-Modal Multimedia Computing

Wireless Communications and Mobile Computing

Generative Adversarial Networks for Multi-Modal Multimedia Computing

Lead Guest Editor: Philippe Fournier-Viger
Guest Editors: Jerry Lin, Rage Uday Kiran, and Yulin Wang

Chief Editor

Associate Editors

Academic Editors

Contents

Hindawi
Wireless Communications and Mobile Computing
Volume 2021, Article ID 5295784, 20 pages
https://doi.org/10.1155/2021/5295784

Research Article

Grid-Based Whole Trajectory Clustering in Road Networks Environment

Fangshu Wang,[1] **Shuai Wang**,[2] **Xinzheng Niu**,[3] **Jiahui Zhu**,[3] **and Ting Chen**[3]

[1] *School of Information and Software Engineering, University of Electronic Science and Technology of China, Chengdu, Sichuan 611731, China*
[2] *School of Information and Communication Engineering, University of Electronic Science and Technology of China, Chengdu, Sichuan 611731, China*
[3] *School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, Sichuan 611731, China*

Correspondence should be addressed to Shuai Wang; wangshuai0601@uestc.edu.cn

Received 5 August 2021; Accepted 22 October 2021; Published 24 November 2021

Academic Editor: Jerry Chun-Wei Lin

In the data mining of road networks, trajectory clustering of moving objects plays an important role in many applications. Most existing algorithms for this problem are based on every position point in a trajectory and face a significant challenge in dealing with complex and length-varying trajectories. This paper proposes a grid-based whole trajectory clustering model (GBWTC) in road networks, which regards the trajectory as a whole. In this model, we first propose a trajectory mapping algorithm based on grid estimation, which transforms the trajectories in road network space into grid sequences in grid space and forms grid trajectories by recognizing and eliminating redundant, abnormal, and stranded information of grid sequences. We then design an algorithm to extract initial clustering centers based on density weight and improve a shape similarity measuring algorithm to measure the distance between two grid trajectories. Finally, we dynamically allocate every grid trajectory to the best clusters by the nearest neighbor principle and an outlier function. For the evaluation of clustering performance, we establish a clustering criterion based on the classical Silhouette Coefficient to maximize intercluster separation and intracluster homogeneity. The clustering accuracy and performance superiority of the proposed algorithm are illustrated on a real-world dataset in comparison with existing algorithms.

1. Introduction

With the advancement of Global Position System (GPS) technology and the growing economy, people's travel is becoming fast and convenient. The massive position and movement information of moving objects are generated continuously, forming large-scale trajectory data. It is of great academic significance and commercial value to mine the underlying distribution information and the evolvement rules, such as urban function partition (Niu et al. [1]), traffic jam prediction (Yu et al. [2]), and privacy protection (Wang et al. [3]). In addition, some classical trajectory clustering algorithms such as DBSCAN (Ester et al. [4]) are widely used in anomaly trajectory detection and anomaly event prevention (Belhadi et al. [5], Belhadi et al. [6], Djenouri et al. [7]). As a significant branch of trajectory data mining, trajectory clustering mainly divides trajectory with high similarity or small distance into one cluster. The main purpose of trajectory clustering is to find representative path or the common moving tendency of different moving objects and extract human behavioral pattern and distribution rules of hot events contained in massive information.

In recent years, a large number of trajectory clustering studies have been published, which can be divided into three categories: trajectory points clustering, subtrajectories clustering, and whole trajectories clustering. Trajectory point clustering partitions are with similar GPS points into the same cluster based on the similarity criteria. Subtrajectory

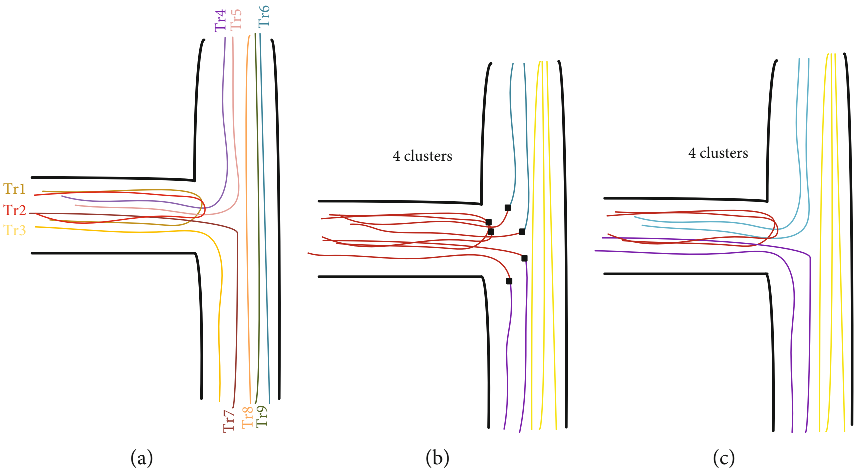


Figure 1: The comparison between subtrajectories clustering and whole trajectories clustering. (a) 9 trajectories in the road network. (b) Partitioning trajectories and subtrajectory clustering result. (c) Whole trajectory clustering result.

clustering firstly divides the whole trajectory into several trajectory segments according to the time stamp, the direction of trajectory points, and road condition and then clusters the trajectory segments based on the similarity between them, while whole trajectory clustering directly forms clusters by calculating the similarity between the whole trajectories. Trajectory point clustering is more applied to extract hotspots. Clustering based on subtrajectories tends to mine movement patterns within a specific period or road segment, but the unit of whole trajectories clustering is much wider in the range of time and space, so this approach can reflect the complete movement trend and rule of trajectories, and the continuous and whole trajectory in the road network shows the connection between the trajectory owner and the external world. That is to say, the whole trajectory clustering can provide more comprehensive support for analyzing different movement patterns in a day and predicting users' next travel information. Figure 1(a) shows 9 trajectories in the road network, which are marked with different colors. Figures 1(b) and 1(c) show the clustering results by the subtrajectory clustering algorithms and the whole trajectory clustering algorithm for the 9 trajectories in Figure 1(a). In both figures, the same color trajectories are in the same cluster. As can be seen from Figures 1(b) and 1(c), these 9 trajectories are clustered by the classic subtrajectory clustering algorithms TraClus [8] and the classic whole trajectory clustering algorithm GridCSD-TraceMob (Han et al. [9]), and they are partitioned into 4 clusters. In Figure 1(b), those marked black squares in some trajectories are the turning points where the trajectories are divided into a series of subtrajectories based on angular offset of adjacent trajectory points and road segments by the algorithm TraClus. In Figure 1(b), it can be observed that the subtrajectories clustering algorithm mines the local trajectory information located in subroad segment but ignores the complete movement rules of objects. However, the whole trajectory clustering treats the trajectory as a complete path, which can better reflect the overall information of the trajectory, as shown in Figure 1(c).

Most of the existing whole trajectories clustering algorithms usually take K-means [10], DBSCAN, and other basic clustering algorithms as the premise and introduce or improve different trajectories distance measurement standards to complete the partition of similar trajectories. Yanagisawa et al. [11] represented the trajectory data as the directed line segments in space and defined the similarity between trajectories as the Euclidean distance between the directed discrete lines, but the algorithm can only compare the trajectories with the same time interval or the same length. To solve this problem, several methods based on warping distance are defined in literatures [12, 13], while Lin and Su [14] propose a method to compare the space shape of trajectory. But these methods still rely on all the position points in trajectories and assume that these points are accurately collected in the road network, while Gariel et al. [15] reorganized the trajectory sequences by identifying the turning points in the trajectory or using the resampling of principal component analysis and the augmented trajectory method and then partitioned similar trajectories based on representative sequences. Although the whole trajectory clustering is well completed in this method, it is not suitable for the vehicle trajectories with very irregular moving process under road network constraints, and the clustering results are limited by the accuracy of the extraction of trajectory representative sequence.

In this paper, the concept of grid cell space is introduced, and a whole trajectory clustering model in road network environment is proposed. Firstly, the sequence of trajectory points in road network space is mapped to grid trajectory based on grid cell space and grid estimation algorithm. Secondly, the center density rule and shape similarity measure are introduced to extract the initial cluster centers, and finally, the outlier function and nearest neighbor principle are combined to dynamically adjust and update the clusters of grid trajectory. The key information of the original trajectory in the road network is retained, and the clustering results with high accuracy are obtained, while solving the problem of large amount of trajectory data and their complex structure.

In sum, our work makes the following technical contributions to the area of trajectory clustering:

(i) A grid cell space is defined for the scattered and changing trajectory data, and an effective mapping algorithm based on grid estimation is designed to transform the complex trajectories in the road network space into the plane grid trajectories in the grid cell space with the original spatial structure preserved

(ii) A clustering algorithm of whole grid trajectories is proposed based on center density rule, shape similarity measure, and anomaly function. The algorithm can accurately identify the abnormal trajectories in the dataset and quickly and efficiently divide the grid trajectories into clusters

(iii) A mapping-clustering-verification framework provides a trajectory clustering analysis model with a Silhouette index-based criterion for clustering performance evaluation.

The rest of this paper is organized as follows. Section 2 conducts a survey of related work. The whole trajectory clustering problem is defined in Section 3. The design of a grid-based whole trajectory clustering model (GBWTC) in road networks is detailed in Section 4. We present experimental results in Section 5 and conclude our work in Section 6.

2. Related Work

Trajectory clustering is mainly divided into point-based clustering, subtrajectories clustering, and whole trajectories clustering. The point-based approaches take GPS points as the basic unit for clustering. The trajectory spatial aggregation pattern is mined and analyzed based on the sparse and dense spatial distribution of vehicle trajectory points, to extract hot area information or key road information. Qiu and Wang [16] improved the structure of the DBSCAN method by combining the position and orientation of the trajectory points, proposed the O-DBSCAN algorithm to divide the entire trajectory point set into representative clusters, and used Gestalt's law to infer the route map. Lu et al. [17] made a breakthrough from the DBSCAN algorithm, redefining the core neighborhood and core objects in the DBSCAN algorithm, introduced a kernel function to measure the similarity between trajectory points, and finally extracted road segments based on the optimized DBSCAN structure information. Yan-Wei et al. [18] further extended the related terminology of DBSCAN, introduced the concept of density through grid cells, converted DBSCAN's extended clustering based on the density of data points to an extended cluster based on the density of cell, and proposed a simple and efficient fast density clustering algorithm CBSCAN. The algorithm can quickly find clustering patterns and noises of arbitrary shapes in location big data. Different from Qiu and Lu, Yu et al. [19] proposed a grid density algorithm based on trajectory points to identify hot spots in different periods of time and used spatiotemporal trajectory clustering methods to mine frequent paths between hot spots. Although the clustering of trajectory points is convenient, concise, and easy to understand, its essence destroys the time continuity of the trajectory. At the same time, it increases the clustering cost due to the similarity calculation of the time complexity of the Cartesian product between the trajectory points in the most of point-based clustering algorithms.

The concept of subtrajectory clustering first appeared in the TraClus [8] algorithm proposed by Lee and Han. The algorithm divides the trajectory into several trajectory segments based on the principle of minimum description length and clusters these trajectory segments based on the DBSCAN algorithm and Euclidean distance. The algorithm has a good effect on hurricane data and animal migration data, but the results have not been very good on real road trajectory datasets, and there are problems such as many clustering parameters and parameter sensitivity. At present, there are a lot of researches to correct these shortcomings, such as the ATCGD algorithm (Mao et al. [20]), NEAT algorithm (Binh Han et al. [21]), and LBTC algorithm (Niu et al. [22]). The ATCGD algorithm maps the divided subtrajectory segments to the grid cell space, then calculates the number of trajectory segments in the grid cell and the distance of the trajectory segments based on this mapping space, adaptively determines the parameters based on the DBSCAN method, and finally completes the clustering. The NEAT algorithm comprehensively considers the speed, flow, density, and other factors of the trajectory. By revising the Hausdorff distance calculation formula, the calculation of the vertical distance, parallel distance, and angular distance between all the line segments of the two subtrajectory sets in the TraClus algorithm is transformed into the distance between the endpoints of two representative trajectories. The flow clusters are combined according to the revised flow distance calculation formula by optimizing the distance between the two flow clusters. However, the road segments are mainly clustered in this algorithm through the traffic flow, and user trajectories are not specifically clustered, so it cannot accurately mine a large amount of user trajectory clustering information. Kumar et al. [23] proposed the Fast-clusiVAT algorithm for this deficiency of NEAT. First, the trajectory is decomposed into a directed graph or an undirected graph. In the process of executing the DTW algorithm, a step is added of using the Dijkstra algorithm to calculate the shortest path between two trajectory segments within a specified range. Moreover, this algorithm can accurately find the trajectory clusters in the dense area of the real road network, but it cannot solve the multidimensional problem of the trajectory. Bermingham and Lee [24] proposed a highly versatile n-dimensional data clustering algorithm and an arbitrary-dimensional representative trajectory extraction algorithm within a cluster, which can cluster any number of trajectory datasets and express valuable, previously unknown higher dimensional trajectory patterns.

In addition, some subtrajectory clustering algorithms expand around subtrajectories, and they generally need to calculate the distance between each point on the subtrajectory and finally add the weights of several different distances. For example, Salarpour and Khotanlou [25] used spectral clustering to segment the trajectory, proposed a trajectory description method based on the change of the subtrajectory direction, and measured the similarity of the described trajectory based on the time warping matching algorithm. Taking into account the uncertainty of trajectory data, Guo et al. [26] proposed a similarity measurement method based on an amended ellipse model, referred to as UTSM, to reduce the interpolation error and positioning error. This method has good robustness and tolerance to abnormal data and noise. In order to clearly describe the difference between the subtrajectories of a moving target, Liu and Zhang [27] proposed a distance measurement method between subtrajectories based on time, space, and direction, but this method ignores the key factor of moving speed. Yu et al. [28] put forward a multifeature subtrajectory similarity measurement method which comprehensively considered the subtrajectory's direction, speed, time, and space location. Trajectory segments are used as the basic unit of similarity evaluation in the subtrajectory clustering algorithm, which reduces the clustering cost to a certain extent and more comprehensively considers the characteristics of trajectory data to accurately identify local differences in trajectories. However, the segmented feature points are difficult to identify and easy to lose, so the subtrajectory clustering algorithm is not good at mining users' complete travel rules, and the clustering result is also easily affected by the segmentation method.

The whole trajectory clustering is to use the whole trajectory as a clustering unit from a more macroperspective, define different whole trajectory similarity evaluation methods according to the scene, and cluster trajectories to mine their information. Domingo-Ferrer and Trujillo-Rasua [29] proposed a trajectory similarity measurement algorithm spatially and temporally and clustered the trajectories through a microaggregation algorithm, while [30] comprehensively considered the spatial, temporal, and shaped characteristics of the trajectories to calculate the similarity between trajectories, and a greedy clustering algorithm is proposed based on this, but it needs to traverse all points in the trajectory to calculate the distance between the trajectories, which consumes more memory. In order to reduce computational cost and improve efficiency, Pan et al. [31] used specific sampling of the complete trajectory and evaluated the similarity between the trajectories based on the sampling points and their density. Experiments show that this method can significantly improve the whole trajectory clustering efficiency while ensuring the accuracy of clustering. The TAD algorithm (Yang et al. [32]) was effective for various complex or special trajectories with long-duration gaps by introducing a noise tolerance fact to evaluate and deal with the influence of noise. Wang et al. [33] proposed a novel vehicle trajectory clustering method based on dynamic network representation learning which can avoid biased results. Stefan et al. [34] proposed a time series distance measurement method MSM based on edit distance, which defines the three steps of move, split, and merge to calculate the cost of mutual conversion between two time series. However, this method is only suitable for simple time series, not for complex or long trajectory series. Yao et al. [35] used a sliding window to extract the movement features of each attribute of the input trajectory and convert it into a feature sequence. The quality representation of each trajectory is obtained by a convolutional neural network, and finally, high-quality trajectory clusters are obtained. However, the automatic encoding of trajectories by deep learning belongs to supervised learning, and it is difficult to be widely used in trajectory data lacking label information. Han et al. [9] proposed the whole trajectory algorithm TRACEMOB, which uses the coincidence rate of the trajectory in the grid as the basis for the trajectory similarity and converts the distance in the grid space into a d-dimensional Euclidean space. Finally, the K-means-based algorithm is used to complete the clustering. But the algorithm does not screen abnormal trajectories or trajectory points and requires secondary mapping before clustering, which is inefficient and easy to cause errors.

The whole trajectory clustering algorithm regards the trajectory as a whole and ensures the integrity of the trajectory compared to the trajectory point clustering and subtrajectory clustering. This algorithm has achieved good results in trajectory clustering. However, the above methods lack effective trajectory preprocessing steps and concise and fast trajectory similarity measurement. In addition, most of them only focus on the accuracy of clustering and neglect their application in the actual road network. For these limitations, we propose a grid-based whole trajectory clustering model in the road network environment, which is aimed at solving the problem of inefficient clustering caused by redundant trajectory points in the road network and inaccurate positioning. Without destroying the internal structure of the trajectory, the complete trajectory is accurately divided into corresponding clusters.

3. Problem Statement

3.1. Trajectory. A trajectory Tr_i of any object in road networks is represented by a list of spatiotemporal points sampled at equal time intervals, denoted as $Tr_i = \langle p_1, p_2, \cdots, p_j, \cdots, p_n \rangle$, where $p_j = ((x_j, y_j), t_j)$ represents the geographic location coordinates (x_j, y_j) of the moving object and t_j is the time stamp recorded when the moving object passes through the location point.

According to the above definition, we formulate the problem of trajectories clustering as follows. Given a set of trajectories $TS = \{Tr_1, Tr_2, \cdots, Tr_p\}$, it is divided into N_C different clusters $C = \{C_1, C_2, \cdots, C_{N_C}\}$. The quality of clustering is usually evaluated by intercluster separation and intracluster homogeneity [36]. In general, a larger intercluster separation and a higher degree of intracluster homogeneity indicate a more accurate clustering. In this work, we adopt the Silhouette Coefficient (SI) which is widely used in clustering validation, to measure the clustering quality of road networks.

The method of Silhouette Coefficient combines the degree of separation and homogeneity to measure the similarity between any trajectory Tr_i and other trajectories of its cluster, and the similarity of other trajectories of different clusters. Specifically, the Silhouette Coefficient is defined as

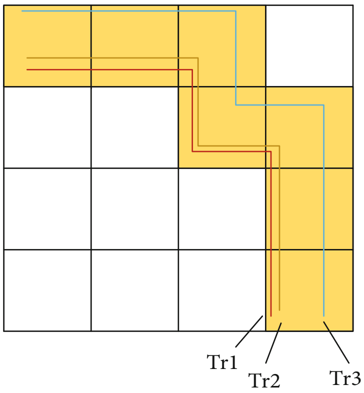


FIGURE 2: Trajectories located in different road segments but close to each other.

$$SI = \frac{1}{N_C}\sum_{i=1}^{N_C}\frac{1}{|C_i|}\sum_{j=1}^{|C_i|}\frac{b(j)-a(j)}{\max(a(j),b(j))}, \quad (1)$$

where $|C_i|$ is the number of trajectories in C_i, $a(j)$ is the average distance between trajectory Tr_j and other trajectories in the same cluster, and $b(j)$ is the minimum average distance between trajectory Tr_j and other clusters. Note that the distance here is calculated by the edit distance of grid trajectories algorithm (*EDG*). The average of the Silhouette Coefficients of all trajectories in TS is the total SI of the clustering result [37]. The value of SI ranges from −1 to 1, and the closer to 1, the better homogeneity and separation are.

We formally define the road trajectory clustering optimization problem as follows:

Given a set of trajectories $\{Tr_1, Tr_2, \cdots, Tr_p\}$, our goal is to process trajectories and abnormal data, to divide the trajectories into groups $\{C_1, C_2, \cdots, C_{N_C}\}$ under the defined similarity criteria, and to output each cluster of trajectories so that the value of SI is close to 1.

In fact, the traditional trajectory clustering based on road networks divides the trajectories located in the same road segment into a group, while ignoring the trajectories located in different road segments but close to each other, as shown in Figure 2. In addition, there may be some subroad segments in a road segment, and the distance between trajectories in the same subroad segment is smaller than that in different subroad segments. However, these trajectories are divided into the same cluster since they all share the same road segment. Therefore, it is unreasonable to cluster vehicle trajectories only based on road segments, so we introduce a whole trajectory clustering model based on grid cell space to solve these problems.

4. A Grid-Based Whole Trajectory Clustering Model: GBWTC

This section will elaborate the proposed grid-based whole trajectory clustering model in road network environment, referred to as GBWTC, from the two stages of grid trajectory serialization and overall clustering algorithm based on grid trajectory. The specific flowchart is shown in Figure 3.

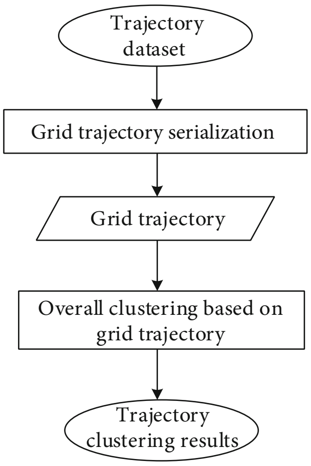

Trajectory dataset
Grid trajectory serialization
Grid trajectory
Overall clustering based on grid trajectory
Trajectory clustering results

FIGURE 3: Overview of our proposed model.

Phase 1 for grid trajectory serialization is as follows: based on the trajectory mapping algorithm, the trajectories in road network space are transformed into grid sequences in grid space, and the redundant, abnormal, and stranded information of grid sequences are eliminated to form the representative sequence of trajectories, i.e., grid trajectories. While retaining the key information of the original trajectory, they can express the moving trend of the trajectories concisely and accurately.

Phase 2 for overall clustering based on grid trajectory is as follows: in the grid trajectory serialization phase, the original trajectory clustering problem in road network is transformed into an overall clustering problem of plane grid trajectory. While K-means is taken as the core in this stage, and the center density rule, shape similarity measure, and outlier function are introduced to deal with the whole clustering of grid trajectories in plane space.

4.1. Grid Trajectory Serialization. Grid trajectory serialization (*gridTrS*) is the process of transforming trajectories from road network space into grid space. This section first gives the following definitions to better describe the process:

Grid space: given a set of trajectories $TS = \{Tr_1, Tr_2, \cdots, Tr_p\}$, the grid space is the minimum boundary rectangle required to cover any trajectory in TS, defined as $R(G) = Rec t(L, W)$, where $L = lat_{\max} - lat_{\min}$ and $W = lon_{\max} - lon_{\min}$, which is the actual geographic coordinate range of the trajectory set TS. $lat_{\max}$ and $lon_{\max}$ are the maximum x and y coordinates of all trajectory points in TS, and $lat_{\min}$ and $lon_{\min}$ are the minimum x and y coordinates of all trajectory points in TS.

Grid cell space: given grid space $R(G)$ and grid cell size *gridsize*, grid cell space is a square with *gridsize* side length in the grid space, defined as

$$S_G(R(G), gridsize) = \{g_{m,n} : 1 \le m \le longrid, 1 \le n \le latgrid, \\ g_{m,n} = Rect(gridsize, gridsize)\}, \quad (2)$$

where $longgrid = \lceil L/gridsize \rceil$, $latgrid = \lceil w/gridsize \rceil$, $\lceil . \rceil$ is rounding up the value, and $g_{m,n}$ denotes a grid cell with $gridsize$ of the length and width in row m and column n of $S_G(R(G), gridsize)$.

Given a set of trajectories $TS = \{Tr_1, Tr_2, \cdots, Tr_p\}$, we first extract grid space $R(G)$ and divide $R(G)$ based on the given $gridsize$ to form grid cell space $S_G(R(G), gridsize)$. Secondly, every trajectory Tr_i in the TS is mapped to grid cell space $S_G(R(G), gridsize)$-based mapping relationship, and each GPS point $p(x, y, t)$ in the trajectory Tr_i falls into the grid cell $g_{m,n}$ corresponding to its position, where the mapping relationship between $p(x, y, t)$ and $g_{m,n}$ is defined as

$$f\left(p(x,y,t), g_{m,n}\right) = \begin{cases} \left\lceil \dfrac{(x - lat_{\min})}{gridsize} \right\rceil, \\ \left\lceil \dfrac{(y - lon_{\min})}{gridsize} \right\rceil, \end{cases} \tag{3}$$

where x,y are the geographic coordinates of point p, $lat_{\min}$ and $lon_{\min}$ are the minimum values of the x and y coordinates of all trajectory points in TS, and $\lceil . \rceil$ is rounding up the value. Any trajectory $Tr_i = \langle p_1, p_2, \cdots, p_k, \cdots, p_h \rangle$ can obtain the corresponding grid cell sequence representing the trajectory points, referred to as $gridTr_i$, i.e., the grid trajectory $gridTr_i = \langle c^i_{m_1,n_1}, c^i_{m_2,n_2}, \cdots, c^i_{m_k,n_k}, \cdots, c^i_{m_h,n_h} \rangle$, where $c^i_{m_k,n_k}$ is the grid cell mapped by the trajectory point p_k on the $gridTr_i$, and m_k, n_k represent the row and column numbers of the grid in $S_G(R(G), gridsize)$, respectively, i is the unique identifier of $gridTr_i$. However, grid trajectory will be redundant, and the clustering cost will increase if several trajectory points are mapped to the same grid cell. Figure 4 shows grid trajectory $gridTr_1 = \langle c^1_{m_1,n_1}, c^1_{m_2,n_2}, \cdots, c^1_{m_{16},n_{16}} \rangle$ mapped by trajectory Tr_1 in grid cell space S_G, i.e., $gridTr_1 = \langle g_{3,4}, g_{3,4}, g_{3,4}, g_{2,4}, g_{2,4}, g_{2,3}, g_{2,3}, g_{2,3}, g_{2,2}, g_{2,2}, g_{2,2}, g_{2,1}, g_{2,1}, g_{1,1}, g_{1,1}, g_{1,1} \rangle$. There are many duplicate grid cells in the grid trajectory because many points map the same grid cell; besides, the trajectory points are easy to drift due to the influence of the collected signals. Therefore, this paper proposes a trajectory mapping algorithm based on grid estimation ($TMGE$). This method combines the characteristics of grid cell space and further identifies and eliminates the abnormal, redundant, and stranded grid cells based on the grid trajectory and completes the data structure optimization of the grid trajectory.

Specifically, $TMGE$ first forms grid trajectories set $GTS = \{gridTr_1, gridTr_2, \cdots, gridTr_p\}$ corresponding to $TS = \{Tr_1, Tr_2, \cdots, Tr_p\}$ based on grid cell space and the mapping relationship. Then, any grid cell $c^i_{m_o,n_o}$ is selected from any grid trajectory $gridTr_i$ in GTS, and its previous grid cell $c^i_{m_{o-1},n_{o-1}}$ in $gridTr_i$ is confirmed if it is the same as $c^i_{m_o,n_o}$. If they are the same, grid cell $c^i_{m_o,n_o}$ is judged as the redundant cell and is removed from the grid trajectory $gridTr_i$. If they are not same, grid cell $c^i_{m_o,n_o}$ is confirmed if it is adjacent to its previous grid cell $c^i_{m_{o-1},n_{o-1}}$ and next grid cell $c^i_{m_{o+1},n_{o+1}}$. If they are not adjacent, i.e., $c^i_{m_o,n_o} \notin Adja(c^i_{m_{o-1},n_{o-1}})$ and $c^i_{m_o,n_o} \notin Adja(c^i_{m_{o+1},n_{o+1}})$, grid cell $c^i_{m_o,n_o}$ is judged as abnormal or outlier cell and is removed from $gridTr_i$, where the set of adjacent grid cells $Adja(c^i_{m_o,n_o})$ of $c^i_{m_o,n_o}$ are the grid cells with the same row(column) as $c^i_{m_o,n_o}$ and whose column (row) spacing difference is 1 with $c^i_{m_o,n_o}$. $Adja(c^i_{m_o,n_o})$ is denoted as

$$\begin{aligned} Adja\left(c^i_{m_o,n_o}\right) = \{ & c_{m_k,n_k}(|m_k - m_o| = 1 \& n_k = n_o) \\ & \cdot \mid (|n_k - n_o| = 1 \& m_k = m_o)\}. \end{aligned} \tag{4}$$

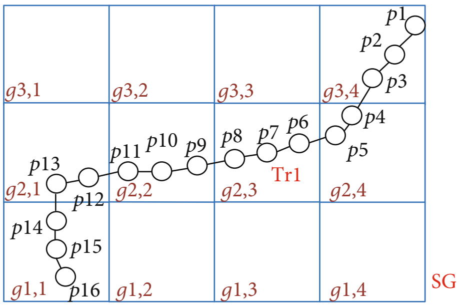
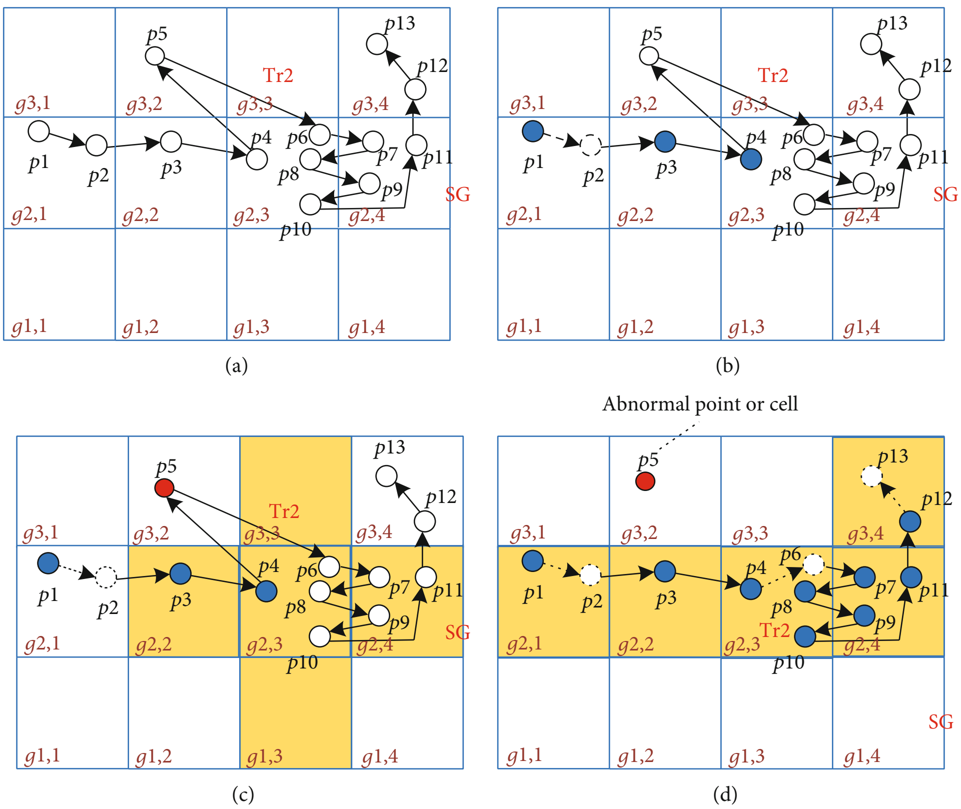


FIGURE 4: Trajectory Tr_1 is mapped to grid cell space.

As is shown in Figure 5, $TMGE$ first maps trajectory $Tr_2 = \langle p_1, p_2, \cdots, p_{13} \rangle$ to grid trajectory $gridTr_2 = \langle c^2_{m_1,n_1}, c^2_{m_2,n_2}, \cdots, c^2_{m_{13},n_{13}} \rangle = \langle g_{2,1}, g_{2,1}, g_{2,2}, g_{2,3}, g_{3,2}, g_{2,3}, g_{2,4}, g_{2,3}, g_{2,4}, g_{2,3}, g_{2,4}, g_{3,4}, g_{3,4} \rangle$. Then, each redundant or abnormal grid cell of $gridTr_2$ is identified and processed. Figure 5(b) shows $c^2_{m_2,n_2}$ mapped by point p_2 is redundant grid cell and is removed from $gridTr_2$, so the point p_2 is drawn with dotted lines. Then, $c^2_{m_3,n_3}$ and $c^2_{m_4,n_4}$ are not identified to be either redundant or abnormal cells, so they are retained in $gridTr_2$. However, as shown in Figure 5(c), $c^2_{m_5,n_5}$ mapped by point p_5 is not adjacent to grid cells $c^2_{m_4,n_4}$ and $c^2_{m_6,n_6}$. Figure 5(d) shows grid trajectory $gridTr_2 = \langle g_{2,1}, g_{2,2}, g_{2,3}, g_{2,4}, g_{2,3}, g_{2,4}, g_{2,3}, g_{2,4}, g_{3,4} \rangle$ by removing the abnormal grid cell $c^2_{m_5,n_5}$ and redundant grid cells $c^2_{m_2,n_2}$, $c^2_{m_6,n_6}$, and $c^2_{m_{13},n_{13}}$ from original $gridTr_2$. In Figure 5(d), the points drawn with dotted lines are redundant, and the points painted with red are abnormal.

However, the subsequence $\langle c^2_{m_4,n_4}, c^2_{m_6,n_6}, c^2_{m_7,n_7}, c^2_{m_8,n_8}, c^2_{m_9,n_9}, c^2_{m_{10},n_{10}} \rangle = \langle g_{2,3}, g_{2,4}, g_{2,3}, g_{2,4}, g_{2,3}, g_{2,4} \rangle$ in $gridTr_2$ moves repeatedly two adjacent cells $g_{2,3}$ and $g_{2,4}$. Therefore, there exist a few stranded cells in this subsequence. As shown in Figure 6(a), only the first and last grid cells in the subsequence are retained, and the other grid cells of continuous repetition are deleted. In Figure 6(a), the points enclosed by the larger ellipse drawn with dotted lines are the removed stranded points, and all deleted points are drawn with dotted lines. Figure 6(b) shows the grid trajectory $gridTr_2 = \langle g_{2,1}, g_{2,2}, g_{2,3}, g_{2,4}, g_{3,4} \rangle$ after processing, the grid trajectory in Figure 6(b) is shown in Figure 6(c) with the background of the highlighted grid cells, and the



(a)

(b)

(c)

(d)

Figure 5: Identification and processing of redundant cells and abnormal cells. (a) The trajectory Tr_2 is mapped to the grid trajectory. (b) Identification of the redundant cell. (c) The adjacent grid cells. (d) Identification and processing of the abnormal cell.

sequence of the highlighted grid cells covering $gridTr_2$ is $\langle g_{2,1}, g_{2,2}, g_{2,3}, g_{2,4}, g_{3,4} \rangle$.

After the trajectory $Tr_i = \langle p_1, p_2, \cdots, p_h \rangle$ in TS is mapped and abnormal, redundant, and stranded grid cells in Tr_i are removed, the final trajectory is denoted by $gridTr_i = \langle c^i_{m_1,n_1}, c^i_{m_2,n_2}, \cdots, c^i_{m_l,n_l} \rangle$, where $l \leq h$.

The pseudocode of the grid trajectory serialization ($gridTrS$) algorithm is presented in Algorithm 1, in which *lines* 2-5 add grid cells mapped by all points of Tr_k in TS to sg_k, *lines* 6-10 remove the redundant and abnormal grid cells from sg_k, *lines* 11-19 remove stranded grid cells and form the eventual grid trajectory $gridTr_k$, and *line* 20 adds $gridTr_k$ to the set of grid trajectories GTS.

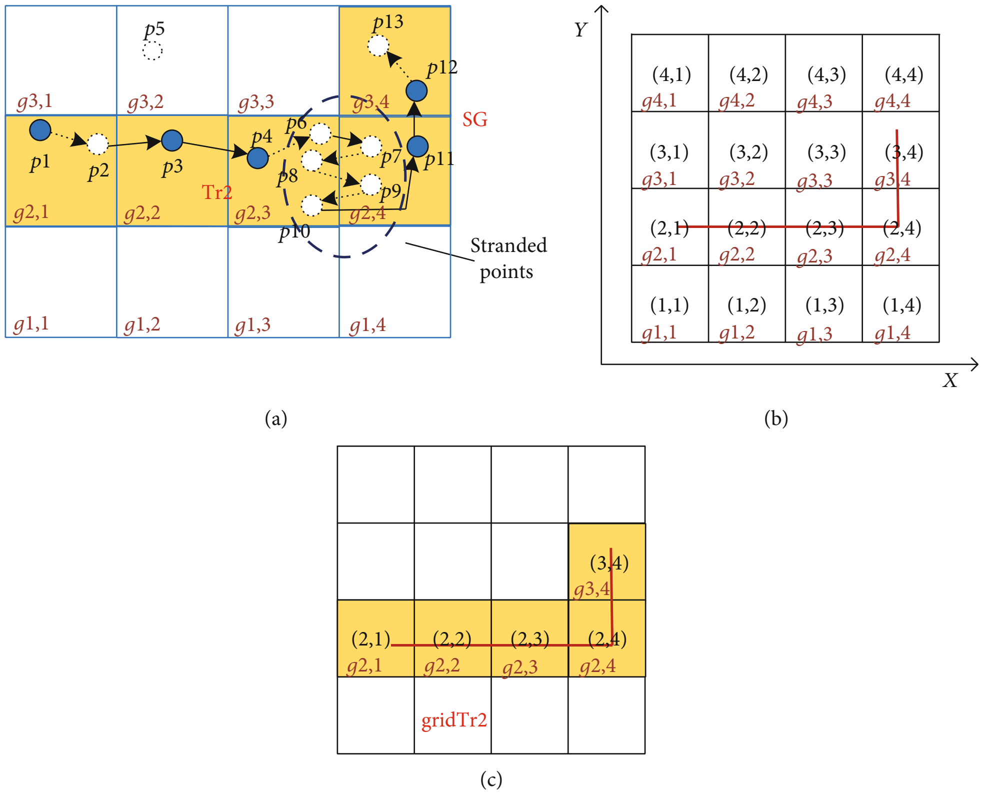
4.2. Overall Clustering Based on Grid Trajectory.

4.2. Overall Clustering Based on Grid Trajectory. As one of the most classic clustering algorithms, K-means is widely used in the trajectory field on account of its simplicity and rapidity, and this algorithm can be completed quickly. However, the algorithm usually uses the trajectory point as the basic unit, so it is not suitable for the whole trajectory clustering. On account of this, this paper proposes an overall clustering algorithm based on grid trajectories with K -means as the core. The algorithm is mainly divided into the formation of initial cluster centers based on density weights and the adjustment and update of clusters based on the grid trajectory.

4.2.1. Formation of Initial Cluster Centers Based on Density Weights. The formation of initial clustering centers based on density weights is to extract the initial cluster centers according to the generated grid trajectory set GTS. The clustering centers of original K-means algorithm are usually selected based on random algorithms. Although it is easy to understand and implement, the random selection of initial clustering mode may result in the clustering results not easy to converge and inconsistent. Therefore, this paper proposes an algorithm to select initial cluster centers ($SICC$). In this algorithm, the distance and density weight concept are introduced to evaluate the probability of grid trajectories becoming cluster centers. Specifically, given a set of grid trajectories $GTS = \{gridTr_1, gridTr_2, \cdots, gridTr_p\}$ and the number of clusters k, $SICC$ first calculates the density $Den(gridTr_i)$ of each grid trajectory $gridTr_i$ in the grid cell space, and the trajectory with the maximum density is selected as the first initial clustering center. The density $Den(gridTr_i)$ is specifically defined as

$$Den(gridTr_i) = \frac{Den\left(c^i_{m_1,n_1}\right) + Den\left(c^i_{m_2,n_2}\right) + \cdots + Den\left(c^i_{m_l,n_l}\right)}{l}, \tag{5}$$

where l is the length of grid trajectory $gridTr_i$ and $Den(c^i_{m_1,n_1})$



(a)

(b)

(c)

Figure 6: Processing of stranded cells and trajectories mapping based on grid estimation. (a) Identification and processing of stranded cells. (b) Trajectories mapping based on grid estimation. (c) The highlighted grid trajectory.

```
Input: $TS = \{Tr_1, Tr_2, \cdots, Tr_p\}$
Output: $GTS = \{gridTr_1, gridTr_2, \cdots, gridTr_p\}$
for each trajectory $Tr_k$ in $TS$ $(1 \le k \le p)$ do
    $sg_k = \phi$;
    $n = sizeof(Tr_k)$;
    for each GPS point $p_l$ in $Tr_k$ $(1 \le l \le n)$ do
        Add $c^k_{m_l,n_l}$ (grid cell mapped by $p_l$) to $sg_k$;
    for each grid cell $c^k_{m_l,n_l}$ in $sg_k$ $(2 \le l \le n)$ do
        if $c^k_{m_l,n_l} = c^k_{m_{l-1},n_{l-1}}$ then
            Remove $c^k_{m_l,n_l}$ from $sg_k$;
        if $c^k_{m_l,n_l} \notin Adja(c^k_{m_{l-1},n_{l-1}})$ & $c^k_{m_l,n_l} \notin Adja(c^k_{m_{l+1},n_{l+1}})$ then
            Remove $c^k_{m_l,n_l}$ from $sg_k$;
    $gridTr_k = \phi$;
    Add $c^k_{m_1,n_1}$ in $sg_k$ to $gridTr_k$;
    for each grid cell $c^k_{m_l,n_l}$ in $sg_k$ $(2 \le l \le sizeof(sg_k))$ do
        $s = sizeof(gridTr_k)$;
        if $s = 1$ then
            if $(c^k_{m_l,n_l}$ in $sg_k) != (c^k_{m_s,n_s}$ in $gridTr_k)$ then
                Add $(c^k_{m_l,n_l}$ in $sg_k)$ to $gridTr_k$;
        if $s1$ then
            if $(c^k_{m_l,n_l}$ in $sg_k) != (c^k_{m_{s-1},n_{s-1}}$ in $gridTr_k)$ & $(c^k_{m_l,n_l}$ in $sg_k) != (c^k_{m_s,n_s}$ in $gridTr_k)$ then
                Add $(c^k_{m_l,n_l}$ in $sg_k)$ to $gridTr_k$;
    Add $gridTr_k$ to $GTS$;
return $GTS$
```

Algorithm 1: *gridTrS*.

```
Input: $GTS = \{gridTr_1, gridTr_2, \cdots, gridTr_p\}$
Output: $Tg\_centers_{ini}$
$Tg\_num \longleftarrow p$ ; $Tg\_dens = \phi$ ; $Tg\_centers = \phi$;
for $i \longleftarrow 1$ to $Tg\_num$ do
    Calculate the density of each grid trajectory: $CalculateTg\_dens(i)$;
sort the vector $Tg\_dens$ in descending order;
$Tg\_centers_{ini}(1) \longleftarrow Tg\_dens(1)$;
for $i \longleftarrow 1$ to $Tg\_num$ do
    For each grid trajectory in $GTS$, using $EDG$ to calculate the distance between it and the selected cluster center: $d_i \longleftarrow EDG($ $Tg\_centers_{ini}(selected), gridTr_i)$;
To sum up all $d_i$: $s_{dis} \longleftarrow sum(d)$; $s_{den} \longleftarrow sum(Tg_dens)$;
$random \longleftarrow (s_{dis}/4) + (s_{den}/4)$;
 while $random > 0$ do
     $random \longleftarrow random - d_i - Tg\_\text{dens}(i)$ ; $//i \in [1, p]$
 $Tg\_centers_{ini}(q) \longleftarrow i$ ; $//q \in [2, k]$
 return $Tg\_centers_{ini}$
```

Algorithm 2: *SICC*.

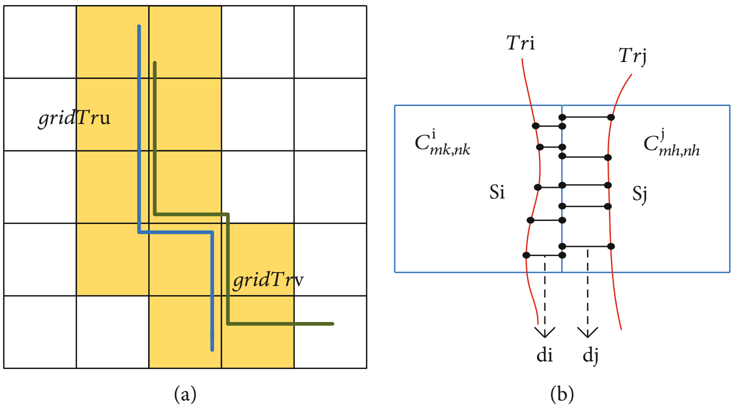
is the density of the grid cell $c^i_{m_1,n_1}$, that is, the number of trajectory points on all trajectories contained in the cell.

Secondly, to follow the principle that the distance between the initial cluster centers is as far as possible and the grid trajectory density of the cluster centers is as large as possible, the distance between the grid trajectories and the cluster centers should be calculated. Since the classical Euclidean distance cannot measure the distance between grid trajectories with different lengths, a new method based on edit distance ([38, 39] is proposed to measure the shape similarity between grid trajectories, referred to as *EDG*. The following concepts are introduced before defining this method.

The insertion cost of grid cells: given two trajectory grids $gridTr_i = \langle c^i_{m_1,n_1}, c^i_{m_2,n_2}, \cdots, c^i_{m_k,n_k}, \cdots, c^i_{m_q,n_q} \rangle$ and $gridTr_j = \langle c^j_{m_1,n_1}, c^j_{m_2,n_2}, \cdots, c^j_{m_h,n_h}, \cdots, c^j_{m_e,n_e} \rangle$, the insertion cost is to insert a grid cell $c^j_{m_h,n_h}$ in $gridTr_j$ into the grid sequence of $gridTr_i$. The cost of the insertion operation is defined as the Euclidean distance between the grid cell $c^j_{m_h,n_h}$ in $gridTr_j$ and the grid cell $c^i_{m_k,n_k}$ being compared in $gridTr_i$. It is denoted as

$$insert\left(c^i_{m_k,n_k}, c^j_{m_h,n_h}\right) = \begin{cases} 0, & isMerging\left(c^i_{m_k,n_k}, c^j_{m_h,n_h}\right) = \text{true}\&k \geq 1, \\ 1, & isMerging\left(c^i_{m_k,n_k}, c^j_{m_h,n_h}\right) = \text{false}\&k1\&h = e, \\ \left\| c^j_{m_h,n_h} \cdot gid - c^j_{m_{h+1},n_{h+1}} \cdot gid \right\|, & isMerging\left(c^i_{m_k,n_k}, c^j_{m_h,n_h}\right) = \text{false}\&k1\&h > 1, \\ \left\| c^j_{m_h,n_h} \cdot gid - c^i_{m_k,n_k} \cdot gid \right\|, & isMerging\left(c^i_{m_k,n_k}, c^j_{m_h,n_h}\right) = \text{false}\&k \geq 1, \end{cases} \quad (6)$$

where e is the length of grid trajectory $gridTr_j$, $c^j_{m_h,n_h} \cdot gid$ represent the row and column numbers of $c^j_{m_h,n_h}$ in S_G, and $||.||$ is the Euclidean distance between them. In this paper, it is transformed into the sum of the absolute value of the difference of each element to improve the efficiency. For example, $||c^j_{m_h,n_h} \cdot gid - c^i_{m_k,n_k} \cdot gid|| = |(m_h - m_k)| + |(n_h - n_k)|$. In fact, the Euclidean distance between the grid cells of different grid trajectories is calculated as the insertion cost, and the grid trajectories in different grid cells have certain distance by default, so the grid trajectories are not considered which are still in the adjacent interval and in the adjacent grid cells. If the case in Figure 7(a) occurs, the distance between the two grid trajectories is very close when transforming $gridTr_u$ to $gridTr_v$. However, there will be a large error in the calculation results due to the high operation costs in different grid cells.



(a)

(b)

Figure 7: Merging of grid cells. (a) The grid trajectories in the adjacent grid cells. (b) The condition of merging.

To solve this problem, $isMerging(c^i_{m_k,n_k}, c^j_{m_h,n_h})$ is introduced to determine whether the grid cells $c^i_{m_k,n_k}$ and $c^j_{m_h,n_h}$ can be merged; the specific calculation formula is as follows:

$$isMerging\left(c^i_{m_k,n_k}, c^j_{m_h,n_h}\right) = \begin{cases} \text{true}, & c^j_{m_h,n_h} \in Adja\left(c^i_{m_k,n_k}\right) \& D\left(s_i, s_j\right) = \dfrac{\sum_{a=0}^{r} d_a^i + \sum_{b=0}^{t} d_b^j}{r + t/2} \le \text{gridsize}, \\ \text{false}, & \text{other}, \end{cases} \tag{7}$$

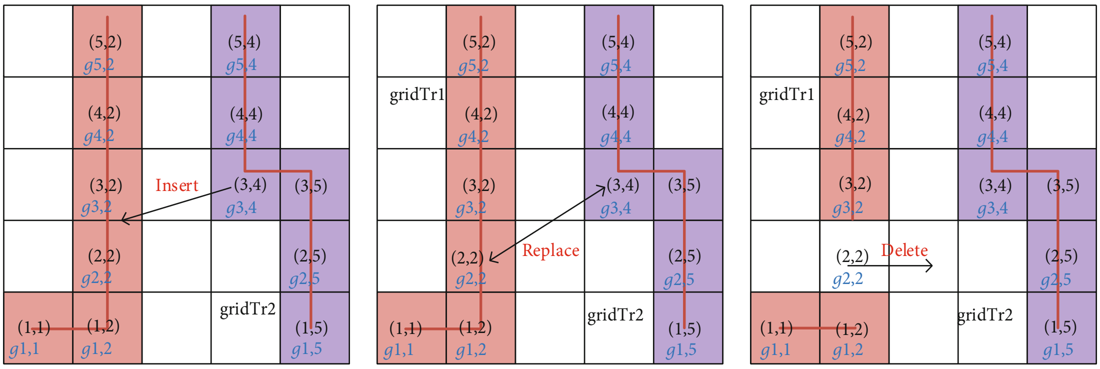
where $c^i_{m_k,n_k}(1 \le k \le q)$ and $c^j_{m_h,n_h}(1 \le h \le e)$ are two grid cells located on $gridTr_i$ and $gridTr_j$ and q and e are the lengths of grid trajectory $gridTr_i$ and grid trajectory $gridTr_j$, respectively. While subsequence $s_i = \langle p_{i_1}, \cdots, p_{i_r}\rangle$ in trajectoy Tr_i is contained in the grid cell $c^i_{m_k,n_k}$, where r is the number of points in s_i, and subsequence $s_j = \langle p_{j_1}, \cdots, \mathrm{p}_{j_t}\rangle$ in trajectoy Tr_j is contained in the grid cell $c^j_{m_h,n_h}$, similarly, t is the number of points in s_j. $\sum_{a=0}^{r} d_a^i$ and $\sum_{b=0}^{t} d_b^j$ are the sum of the vertical Euclidean distances from each location point on subsequences s_i and s_j to the coincident boundaries of grid cell $c^i_{m_k,n_k}$ and $c^j_{m_h,n_h}$. If the value of distance $D(s_i, s_j)$ exceeds gridsize, i.e., the size of the grid cell, the calculation will be terminated. Subsequences s_i and s_j can be regarded as located in one grid cell if the grid cells can be merged, as shown in Figure 7(b).

The replacement cost of grid cells: given two trajectory grids $gridTr_i = \langle c^i_{m_1,n_1}, c^i_{m_2,n_2}, \cdots, c^i_{m_k,n_k}, \cdots, c^i_{m_q,n_q}\rangle$ and $gridTr_j = \langle c^j_{m_1,n_1}, c^j_{m_2,n_2}, \cdots, c^j_{m_h,n_h}, \cdots, c^j_{m_e,n_e}\rangle$, the replacement cost is to transform a grid cell $c^i_{m_k,n_k}$ specified in $gridTr_i$ to a grid cell $c^j_{m_h,n_h}$ in $gridTr_j$. The cost of the replacement operation is defined as the Euclidean distance between $c^i_{m_k,n_k}$ and $c^j_{m_h,n_h}$. It is denoted as

$$replace\left(c^i_{m_k,n_k}, c^j_{m_h,n_h}\right) = \begin{cases} 0, & \text{is}\,Merging\left(c^i_{m_k,n_k}, c^j_{m_h,n_h}\right) = \text{true}, \\ \left\| c^j_{m_h,n_h} \cdot gid - c^i_{m_k,n_k} \cdot gid \right\|, & \text{is}\,Merging\left(c^i_{m_k,n_k}, c^j_{m_h,n_h}\right) = \text{false}, \end{cases} \tag{8}$$

where the replacement cost between grid cells $c^i_{m_k,n_k}$ and $c^j_{m_h,n_h}$ is 0 if the grid cells merging condition in Equation (7) is satisfied.

The deletion cost of grid cells: given two trajectory grids $gridTr_i = \langle c^i_{m_1,n_1}, c^i_{m_2,n_2}, \cdots, c^i_{m_k,n_k}, \cdots, c^i_{m_q,n_q}\rangle$, the deletion cost is to delete a grid cell specified in $gridTr_i$.The cost of the deletion operation is defined as the Euclidean distance between the current grid cell $c^i_{m_k,n_k}$ to be deleted and the next uncompleted cell $c^i_{m_{k+1},n_{k+1}}$ in the grid sequence of $gridTr_i$. It is denoted as



(a) (b) (c)

Figure 8: Transformation between grid trajectories. (a) The insertion of grid cell. (b) The replacement of grid cells. (c) The deletion of grid cell.

$$delete\left(c^i_{m_k,n_k}\right) = \begin{cases} 1, & k = q, \\ \left\| c^i_{m_k,n_k} \cdot gid - c^i_{m_{k+1},n_{k+1}} \cdot gid \right\|, & 1 < k \le q. \end{cases} \tag{9}$$

Figure 8(a) shows the grid trajectory $gridTr_1 = \langle c^1_{m_1,n_1}, c^1_{m_2,n_2}, \cdots, c^1_{m_6,n_6} \rangle = \langle g_{5,2}, g_{4,2}, g_{3,2}, g_{2,2}, g_{1,2}, g_{1,1} \rangle$ and $gridTr_2 = \langle c^2_{m_1,n_1}, c^2_{m_2,n_2}, \cdots, c^2_{m_6,n_6} \rangle = \langle g_{5,4}, g_{4,4}, g_{3,4}, g_{3,5}, g_{2,5}, g_{1,5} \rangle$. Suppose $c^1_{m_4,n_4}$ in $gridTr_1$ and $c^2_{m_3,n_3}$ in $gridTr_2$ are compared. If the grid cell $c^2_{m_3,n_3}$ is inserted in front of $c^1_{m_4,n_4}$ in $gridTr_1$, the insertion cost is $|(3-2)|+|(4-2)| = 3$. Figure 8(b) shows that the grid cell $c^1_{m_4,n_4}$ in $gridTr_1$ is replaced by $c^2_{m_3,n_3}$, and the cost of the replacement operation is $|(3-2)|+|(4-2)| = 3$. Figure 8(c) shows that $c^1_{m_4,n_4}$ is removed from the grid cell sequence of $gridTr_1$, and the deletion cost is $|(2-1)|+|(2-2)| = 1$.

To sum up, the edit distance from grid trajectory $gridTr_i$ to $gridTr_j$ is the sum of the operation costs of insertion, replacement, and deletion; $EDG(gridTr_i, gridTr_j)$ is defined as

$$EDG(gridTr_i, gridTr_j) = \begin{cases} \sum_{h=1}^{e} insert\left(c^i_{m_k,n_k}, c^j_{m_h,n_h}\right), & q = 0, \\ \sum_{k=1}^{q} delete\left(c^i_{m_k,n_k}\right), & e = 0, \\ \min\left\{EDG(Rest(gridTr_i), Rest(gridTr_j)) + replace\left(c^i_{m_k,n_k}, c^j_{m_h,n_h}\right), EDG(gridTr_i, Rest(gridTr_j)) + insert\left(c^i_{m_k,n_k}, c^j_{m_h,n_h}\right), EDG(Rest(gridTr_i), gridTr_j) + delete\left(c^i_{m_k,n_k}\right)\right\}, & otherwise, \end{cases} \tag{10}$$

where $c^i_{m_k,n_k}$ $(1 \le k \le q)$ is the kth grid cell in the trajectory sequence of $gridTr_i$ and $c^j_{m_h,n_h}$ is the hth grid cell in the trajectory sequence of $gridTr_j$. $Rest(gridTr_i)$ is defined as the grid cells other than the compared grid cells in $gridTr_i$, similarly, $Rest(gridTr_j)$ is defined as the grid cells other than the compared grid cells in $gridTr_j$. From formula (10), it can be concluded that the higher the value of edit distance between two grid trajectories, the more dissimilar they are, otherwise, the higher the degree of similarity.

Finally, based on the first initial cluster center that has been determined, $k-1$ centers of grid trajectory cluster with larger distance and higher density are selected. A random value $random$ is set to fuse the density and distance of grid trajectories, denoted as

$$random = \begin{cases} \dfrac{s_{den} + s_{dis}}{b}, & (a) \\ random - EDG(Tg\_centers_{ini}(selected), gridTr_i) - D(gridTr_i) i \in [1, p] \& i! = m, & (b) \end{cases} \tag{11}$$

```
Input: $gridTr_1 = \langle c^1_{m_1,n_1}, c^1_{m_2,n_2}, \cdots, c^1_{m_N,n_N} \rangle$, $gridTr_2 = \langle c^2_{m_1,n_1}, c^2_{m_2,n_2}, \cdots, c^2_{m_M,n_M} \rangle$
Output: $EDG(gridTr_1, gridTr_2)$
Initialize the zeroth row and column of the $N+1$ rows and $M+1$ columns distance matrix $D$: $D[0,0] \longleftarrow 0$;
for each row $i \longleftarrow 1$ to $N$ do
    $D[i,0] \longleftarrow D[i-1,0] + delete(c^1_{m_i,n_i})$;
for each column $j \longleftarrow 1$ to $M$ do
    $D[0,j] \longleftarrow D[0,j-1] + insert(c^1_{m_0,n_0}, c^2_{m_j,n_j})$;
for each row $i \longleftarrow 1$ to $N$ do
    for each column $j \longleftarrow 1$ to $M$ do
        $D[i,j] = \min\{D[i-1,j-1] + replace(c^1_{m_i,n_i}, c^2_{m_j,n_j}), D[i,j-1] + insert(c^1_{m_i,n_i}, c^2_{m_j,n_j}), D[i-1,j] + delete(c^1_{m_i,n_i})\}$;
return $D[N,M]$
```

Algorithm 3: *EDG*.

```
Input: $GTS = \{gridTr_1, gridTr_2, \cdots, gridTr_p\}$, Predefined number of trajectory clusters $k$, The initial clustering centers $Tg\_centers_{ini}$
Output: $TgClusters, TgAbnormal$
Initialize the parameter $k$;
$iter \longleftarrow 0$;
$Tg\_Centers \longleftarrow Tg\_centers_{ini}$;
while $Tg\_centerschanged \mid iter \leq 10$ do
    $iter \longleftarrow iter + 1$ ; $TgClusters = \phi$ ; $TgAbnormal = \phi$;
    for $i \longleftarrow 1$ to $p$ do
        if $i \notin Tg\_centers$ then
            $d_i \longleftarrow \min\ (EDG(Tg\_Centers, gridTr_i))$ and record the index $q$ of $Tg\_Centers$;
            if $R(gridTr_i, TgClusters\{q\}) \geq 0$ then
                $TgClusters\{q\} \longleftarrow TgClusters\{q\} \cup \{i\}$;
            else
                $TgAbnormal \longleftarrow TgAbnormal \cup \{i\}$;
    for $h \longleftarrow 1$ to $k$ do
        $numc \longleftarrow length(TgClusters\{h\})$
        for $i \longleftarrow 1$ to $numc$ do
            for $j \longleftarrow i$ to $numc$ do
                $dsum(i) \longleftarrow dsum(i) + dist(i,j)$;
            if $dsum(i) < \min\ \{dsumin\}$ then
                $dsumin \longleftarrow dsum(i)$;
                $ClustID \longleftarrow i$;
        $Tg\_centers(h) \longleftarrow ClustID$;
return $TgClusters, TgAbnormal$
```

Algorithm 4: *GBWTC*.

where s_{den} is the sum of the density of the grid trajectories other than clustering centers. $EDG(Tg\_centers_{ini}(selected), gridTr_i)$ is the distance between the grid trajectory just selected as the clustering center $Tg\_centers_{ini}(selected)$ and the grid trajectory of non clustering centers $gridTr_i$, and s_{dis} is the sum of these distances, that is, $s_{dis} = sum(EDG(Tg\_centers_{ini}(selected), gridTr_i))$. m is the index of the grid trajectory just selected as the clustering center in *GTS*. In formula (11)(a), the value of *random* is initialized, and after repeated experiments, the value of b is set as 4; after assigning the initial value to *random*, the formula (11)(b) is executed until the value of *random* is less than 0, and the grid trajectory is the next selected cluster center at this time. The above steps are repeated until all initial cluster centers are selected.

4.2.2. The Adjustment and Update of Clusters Based on the Grid Trajectory. The adjustment and update of clusters based on the grid trajectory are a process of dynamically allocating the optimal clusters of grid trajectories according to the nearest neighbor principle. After determining the cluster centers, the traditional K-means method divides each trajectory into the cluster which is closest to the trajectory, but there is no determination of the abnormal trajectories in the trajectory dataset in the iteration process. For this problem, an outlier function $R(gridTr_i, C_a)$ is introduced to measure the influence of grid trajectory on other trajectories in cluster and determine whether the grid trajectory is abnormal. If it is not abnormal, it will be added to the cluster; if it is, the grid trajectory is added to the abnormal trajectory set. The outlier function is specifically defined as

$$R(gridTr_i, C_a) = \begin{cases} d_{c\_\min} - EDG(gridTr_i, gridTr_c), & |C_a| = 1, \\ \left(\dfrac{1}{|C_a| - 1} \sum_{gridTr_j \in C_a} EDG\left(gridTr_j, gridTr_c\right) \right) - EDG(gridTr_i, gridTr_c), & |C_a| > 1, \end{cases} \quad (12)$$

where $d_{c\_\min}$ is the minimum distance between cluster centers. C_a represents the cluster with the nearest cluster center to the grid trajectory $gridTr_i$, and $gridTr_c$ is the center of the cluser C_a. $|C_a|$ is the number of grid trajectories contained in cluster C_a.

Specifically, the outlier function first calculates the distance between the grid trajectory and the cluster center and finds the nearest cluster center. Then, based on the average distance between all the other grid trajectories in the cluster and the cluster center, the influence of the trajectory on the existing structure of the cluster is numerically calculated. As shown in Equation (12), generally, the smaller the value of $R(gridTr_i, C_a)$ is, the smaller the influence of judging the grid trajectory as abnormal. If the value of $R(gridTr_i, C_a)$ is less than 0, the grid trajectory will be marked as abnormal.

The pseudocode of the grid-based whole trajectory clustering GBWTC is presented in Algorithm 4. There are mainly two steps in the algorithm. Firstly, the related variables are initialized (*lines* 1-2), and the initial clustering centers determined by Algorithm 2 are assigned to the clustering centers (*line* 3). Then, the clusters are adjusted and updated based on the iterative process, and the clustering centers are updated after each iteration. If the conditions are met, the clustering process will stop, and the trajectory clusters and abnormal trajectory set will be output (*lines* 4-22). Specifically, according to the outlier function, it is determined whether each trajectory can be clustered into a cluster or temporarily as an exception (*lines* 4-12). The new trajectory centers are calculated according to the clustering results. The clustering process is terminated until the end condition of the iterative process is satisfied (*lines* 14-21). Finally, the algorithm GBWTC outputs a set of trajectory clusters TgClusters and a set of abnormal trajectories TgAbnormal (*lines* 22).

5. Experimental Evaluation

5.1. Experimental Data. We use two real-world datasets as experimental data to verify the efficiency and accuracy of the algorithm: (i) travel records of 536 taxis in San Francisco in more than 30 days that include the longitude and latitude of the vehicles, vehicle ID, time stamp, and whether they carry passengers or not. In this paper, we extract the data of a car in a day according to the time stamp, including 11943 trajectories and filter the longitude and latitude that do not belong to San Francisco city. Finally, we select the longitude and latitude, sampling time, and other attributes to participate in the experiment; (ii) a month's driving data of 320 taxis in Rome City that includes the longitude and latitude of the vehicle, vehicle ID, time stamp, and other information. The dataset is also preprocessed according to the above method, and a total of 7356 trajectories are obtained eventually. Table 1 and Figure 9 show the statistical information of the above two trajectory datasets and the road network composed of the trajectory.

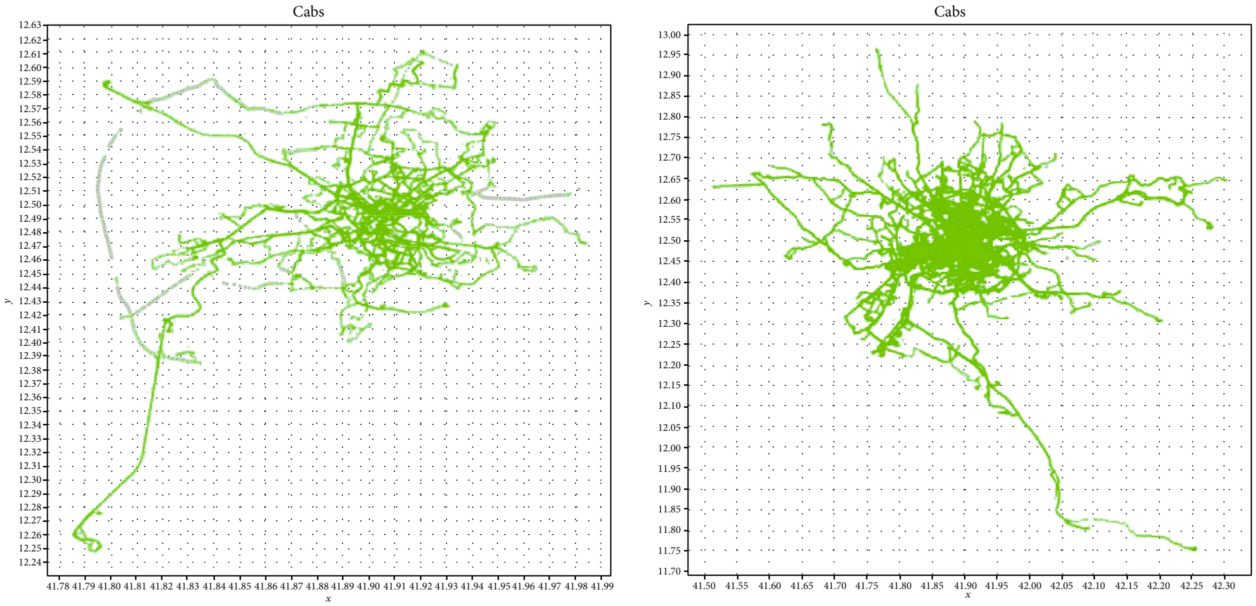
Figure 10 plots the initial distribution of users by average travel time per day in San Francisco and Roman. It can be observed that 89.23% and 82.79% of users have more than 6 hours of travel time in a day in the two datasets, which provides abundant trajectory data for the experiment. There are differences in more subdivided periods, as shown in Figures 10(a) and 10(b), and 39.34% of users travel 6-9 hours a day in the San Francisco dataset, while 68.56% users travel 6-9 hours a day in the Roman dataset. The existence of difference increases the diversity and persuasiveness of experimental results. Further, we plot the mean and standard deviation of the number of users each day per week in the San Francisco and Roman datasets as showed in Figure 11. It shows the stability of the San Francisco data and the Rome data over time. The box plots of normalized longitude and latitude of all moving trajectories of users each week in a month in two datasets are shown in Figure 12. It can be observed that the movement of users is basically in the same region, and there are some differences in the range of motion. Therefore, the data are valuable and the results are representative.

5.2. Raw Algorithms in Comparison. GridCSD-TraceMob firstly executes the trajectory mapping algorithm to calculate the distance between trajectories iteratively. Each iteration is divided into three steps: (1) the two trajectories with the largest distance are selected as the initial pivots, (2) each trajectory is mapped to d-dimensional metric space, and (3) the Euclidean distances between trajectories in the space are calculated. After the iteration, the classical K-means algorithm is used for clustering.

TRACLUS partitions a trajectory into a series of subtrajectories and performs DBSCAN to group similar subtrajectories together. The algorithm determines whether each

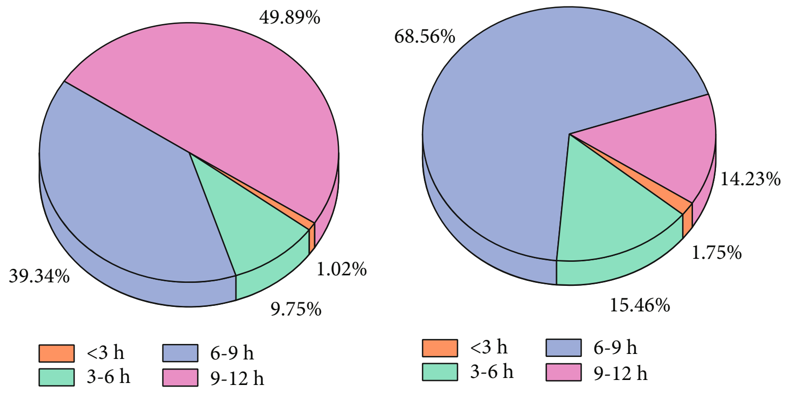
Table 1: Dataset statistics.

Regions	Memory of dataset	Acquisition time	Number of trajectories	Number of GPS points	Sampling interval
San Francisco Bay Area	411 MB	2008.5.17–6.10	11943	12787048	10 s
Roman	1.49 GB	2014.2.1–3.2	7356	21817852	7 s



(a) (b)

Figure 9: The distribution of real datasets. (a) San Francisco Bay Area. (b) Roman.



(a) (b)

Figure 10: Initial distribution of users by average travel time. (a) San Francisco datasets. (b) Roman datasets.

point of the trajectory meets the segmentation conditions based on the minimum description principle. In the DBSCAN phase, the iteration is executed for n_{sub} times, which is the number of subtrajectories. In each iteration, the subtrajectory in the cluster is performed two steps: (1) ε-neighborhood query and (2) cluster expansion performs a linear scanning for each selected subtrajectory's neighbors.

K-means is a classical clustering method, in which clusters are groups of elements characterized by a small distance to the cluster center. The general process of K-means is to assign each element to the nearest cluster center and update the cluster centers. This process is repeated until a convergence condition is satisfied.

5.3. Parameter Settings. The parameter *gridsize* is as follows. To determine the sequence of trajectory grid cells, the size of grid cell *gridsize* needs to be provided to grid cell space. Users may provide their desirable parameters or use the suggested parameters. Generally, most of the dense road sections are concentrated in the city center in the whole dataset of road network, while the distribution of trajectory in some suburban or marginal areas is relatively sparse. Hence, the

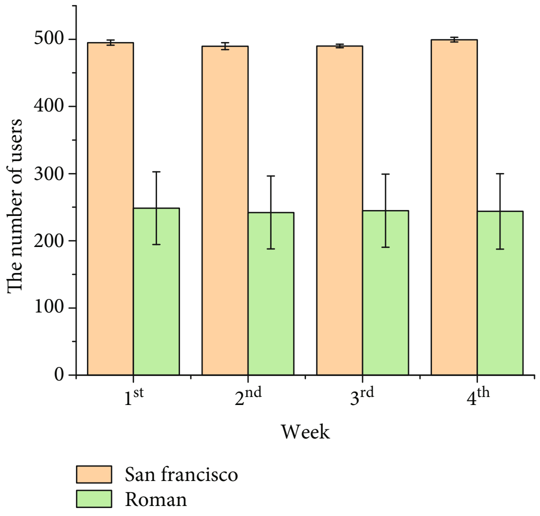
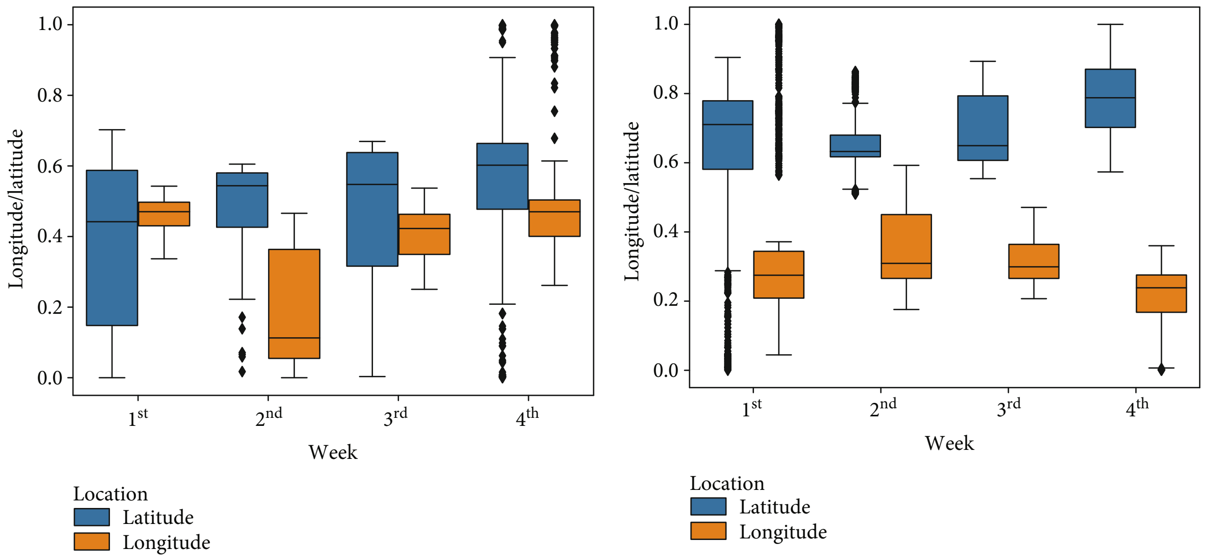


FIGURE 11: The average number of users each day per week.



(a)

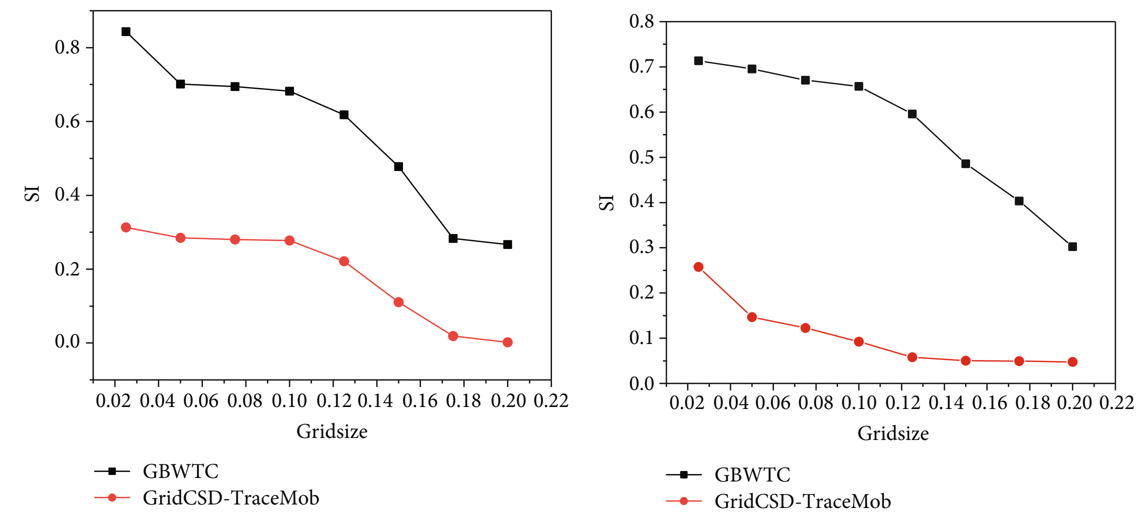
(b)

FIGURE 12: The box plots of the longitude and latitude distribution. (a) San Francisco datasets. (b) Roman datasets.

size of grid cell in grid cell space should not be set too large, to avoid the loss of vehicle driving conditions in dense road sections. But it should not also be set too small; otherwise, the efficiency of the subsequent trajectory clustering stage will be reduced. In our experiments, the grid cell size is set to be 0.1 by default.

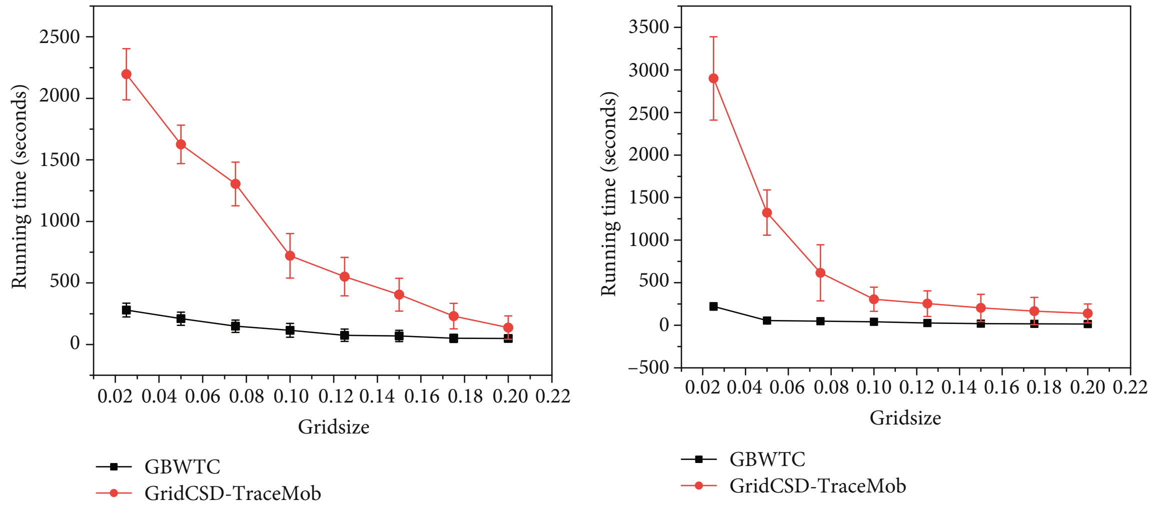
In our experiments, to identify the optimal grid cell size, we run GBWTC and GridCSD-TraceMob algorithms with different grid cell sizes ranging from 0.025 to 0.2 at an interval of 0.025 on the Roman dataset and San Francisco dataset. Figure 13 shows the performance comparison of these two clustering algorithms in terms of SI under given trajectory number and cluster number on the datasets, respectively. As shown in Figures 13(a) and 13(b), the SI index of both decreases gradually with the increase of *gridsize*. The reason is that the trajectories located in different road segments or far away from each other will be divided into the same grid when *gridsize* is set too large, which will cause the error of trajectory similarity measurement. Under the same *gridsize* , even when *gridsize* is larger, the SI index obtained by GBWTC is higher in two different road network datasets, which indicates that the trajectory clustering results obtained by GBWTC have higher similarity in the same cluster, and the separation degree of trajectories between different clusters is also higher, that is, compared with GridCSD-Tracemob, GBWTC has a better overall clustering quality.

However, the efficiency of the algorithm affected by *gri dsize* cannot be taken into account only by the SI index, and the optimal parameters cannot be determined. We further compare GBWTC with GridCSD-TraceMob in terms of running time with different *gridsize* ranging from 0.025 to 0.2 at an interval of 0.025. We plot the mean and standard deviation of the algorithm running time across 8 grid cell spaces with different sizes in Figure 14. As illustrated in



(a) (b)

Figure 13: Comparison of clustering quality using SI with different *gridsize*. (a) San Francisco datasets (2400 trajectories, $k = 15$). (b) Roman datasets (2000 trajectories, $k = 15$).



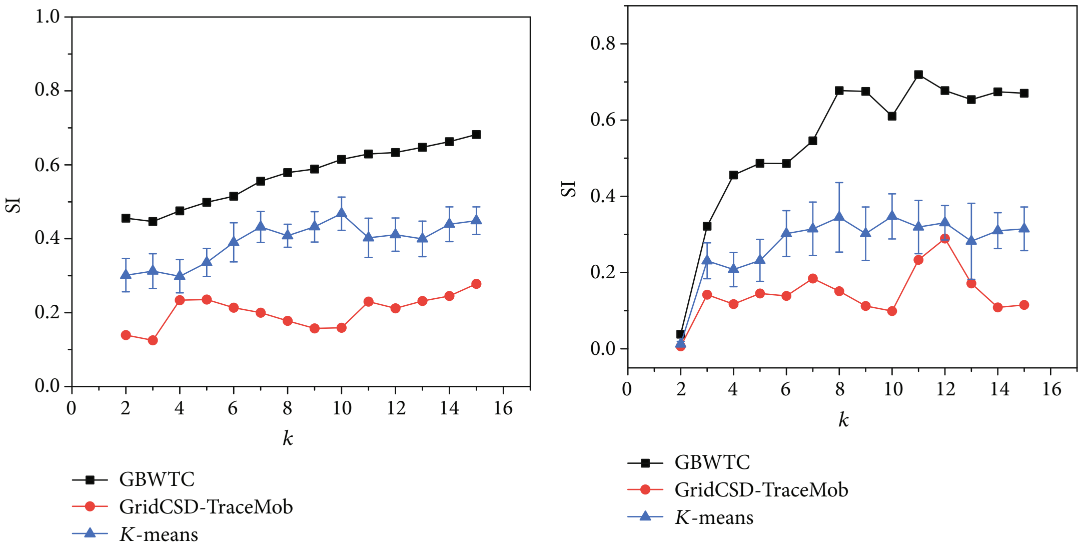
(a) (b)

Figure 14: Comparison of algorithm running time with different *gridsize*. (a) San Francisco datasets (2400 trajectories, $k = 15$). (b) Roman datasets (2000 trajectories, $k = 15$).

Figures 14(a) and 14(b), the running time presents an overall downward trend with the increase of *gridsize*. The larger the *gridsize*, the shorter the running time. It is obvious that the running time of GBWTC is shorter and smoother than that of GridCSD-TraceMob algorithm, and the standard deviation of running time is smaller. Moreover, we observe that GBWTC is approximately four times faster than GridCSD-TraceMob on average.

It can be observed from Figures 13 and 14 that the SI index is higher and the running time is faster than other smaller grid cell sizes when *gridsize* is 0.1, which is a more suitable *gridsize* parameter. Therefore, the *gridsize* is set as 0.1 of GBWTC and GridCSD-TraceMob in the other experiments of this paper.

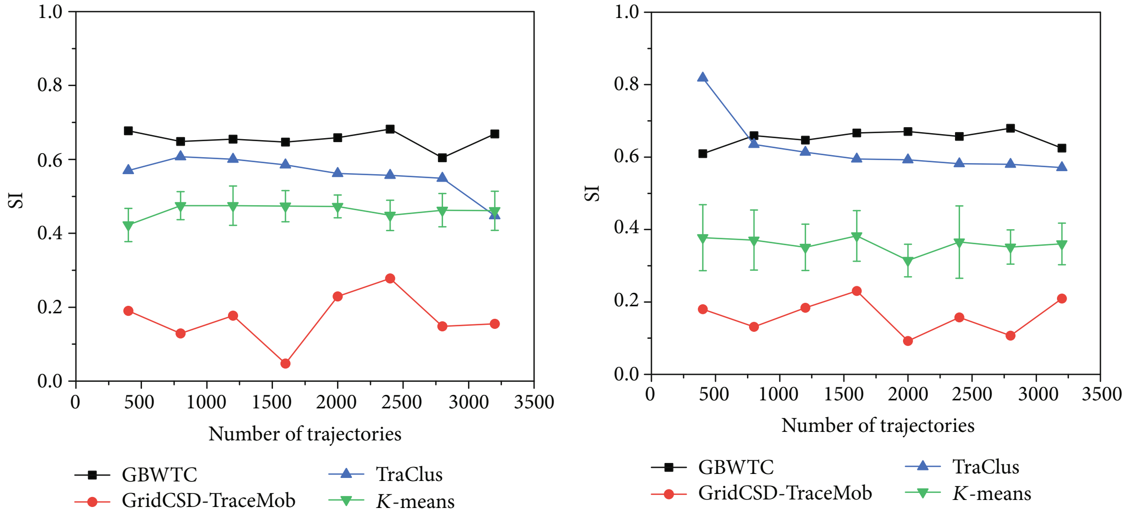
The parameter k is as follows. To better divide the trajectories, the number of clusters k needs to be provided. In our work, we run GBWTC, GridCSD-TraceMob, and K-means algorithms with different numbers of clusters ranging from 2 to 15 at an interval of 1. Since TraClus is a density-based clustering algorithm, it does not participate in the comparison. Figure 15 plots the mean and standard deviation of SI index of the three algorithms across 14 different numbers of clusters in the San Francisco and Roman datasets. For these two datasets, no matter how the value of k changes, GBWTC shows a stronger clustering effect than the other two algorithms, which shows the effectiveness of our improvement in the distance measurement and the steps in the



(a)

(b)

FIGURE 15: Comparison of clustering quality using SI with different numbers of clusters. (a) San Francisco datasets (2400 trajectories, *gri dsize* = 0.1). (b) Roman datasets (2000 trajectories, *gridsize* = 0.1).



(a)

(b)

FIGURE 16: Comparison of clustering quality using SI with different numbers of trajectories. (a) San Francisco datasets ($k = 15$, *gridsize* = 0.1). (b) Roman datasets ($k = 15$, *gridsize* = 0.1).

clustering process to some extent. Since three algorithms have good clustering effect with 15 clusters in the two datasets, the number of clusters k of GBWTC, GridCSD-TraceMob, and K-means is set by 15 in all other experiments of this paper.

5.4. Performance Evaluation. We conduct a simulation-based evaluation of our proposed grid-based whole trajectory clustering model in terms of clustering quality and runtime performance in comparison with existing approaches. We implement all of these algorithms in IntelliJ IDEA 2018.3.5 (64-bit), and all experiments are conducted on a Windows PC workstation equipped with Intel(R) Core(TM) i5-5200U CPU@2.20 GHz and 4 GB of memory.

We evaluate the clustering quality of GBWTC in comparison with GridCSD-TraceMob, TraClus, and K-means in terms of Silhouette Coefficient, which are representatives of whole trajectory-based, subtrajectory-based, and point-based clustering. The value of Silhouette Coefficients is proportional to the clustering quality, and the closer to 1, the better the clustering quality is. We run these four algorithms, GBWTC, GridCSD-TraceMob, TraClus, and K-Means, with different numbers of trajectories ranging from 400 to 3200 at an interval of 400. For each number of trajectories, we generate 8 random trajectory datasets. Figure 16 plots the mean and standard deviation of the SI index across 8 different datasets for each number of trajectories in San Francisco and Roman datasets. It can be observed that the SI index

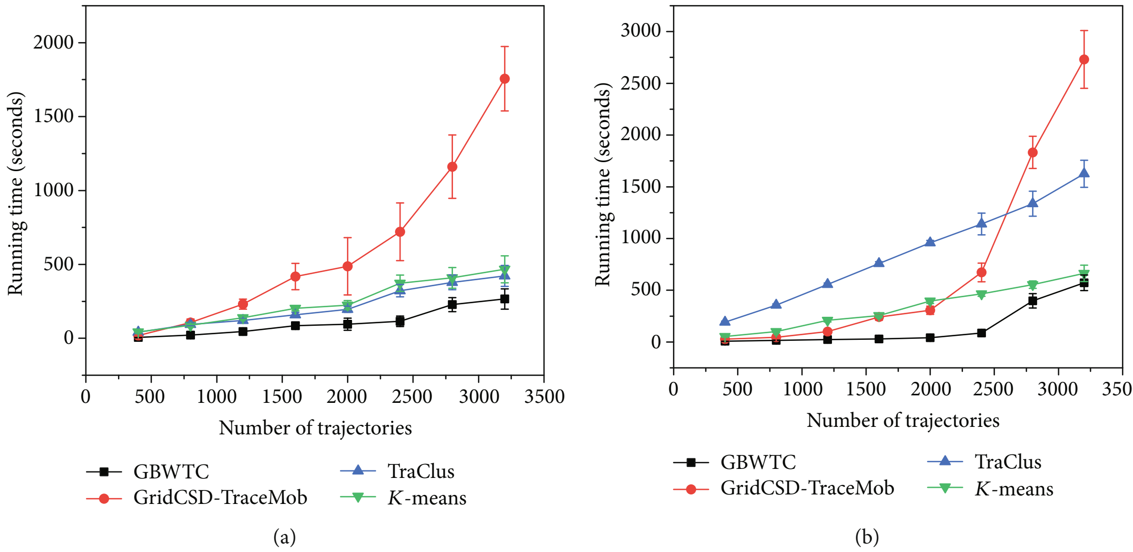


Figure 17: Comparison of algorithm running time with different numbers of trajectories. (a) San Francisco datasets ($k = 15$, $gridsize = 0.1$). (b) Roman datasets ($k = 15$, $gridsize = 0.1$).

Table 2: Comparison of running time.

Number of trajectories	20	40	60	80	100	120
Running time of Scheme 1 (s)	0.33	0.72	0.87	0.99	1.12	1.26
Running time of Scheme 2 (s)	37.27	68.62	87.71	122.76	157.29	184.95

of GBWTC algorithm is closer to 1. With the change of parameters, the SI index of GBWTC is higher than that of the other three clustering algorithms in most cases, that is to say, it shows good adaptability and effect in dealing with different number of trajectories.

Figure 17 compares the proposed clustering algorithm GBWTC with GridCSD-TraceMob, TraClus, and K-means in terms of running time. We run these four algorithms with different numbers of trajectories on the real-world San Francisco and Roman datasets, ranging from 400 to 3200 at an interval of 400 trajectories. Again, for each number of trajectories, we generate 8 random trajectory datasets. We plot the mean and standard deviation of the algorithm running time across 8 datasets for each number of trajectories in Figure 17. These results show that GBWTC runs significantly faster than the other three algorithms in comparison for both San Francisco and Roman datasets, and the change speed of GBWTC algorithm is more gentle. The superiority of GBWTC becomes more obvious as the number of trajectories increases. This is because the GBWTC algorithm eliminates some useless points in the dataset before clustering and optimizes the selection of the clustering center in the clustering process, so that the clustering process is easier to converge, and there is no need for secondary mapping in the clustering process.

In addition, there are two advantages in the implementation of the steps of deleting redundant, abnormal, and stranded cells in the trajectory grid serialization: one is to reduce the running time; the other is to reduce the interference of these cells on clustering results. Table 2 shows the time comparison of whether to remove redundant, abnormal and stranded cells in San Francisco dataset using the GBWTC algorithm. The first scheme is to remove the redundant points, while the second scheme is not.

6. Conclusion

We proposed a novel grid-based whole trajectory clustering model, referred to as GBWTC, which leverages the mapping theory to form the simple and representative grid trajectory. The proposed approach has potential to determine not only a series of trajectory clusters but also some abnormal trajectories and GPS points. Extensive experiments demonstrated that GBWTC significantly improves the clustering quality over the existing methods. The proposed whole trajectory clustering approach has a wide range of applications in various traffic and location service systems, including vehicle path planning, urban planning, service ecommendation, traffic navigation, logistics and distribution, and detection and prevention of abnormal events.

Data Availability

The San Francisco Bay Area Dataset analyzed during the current study is available in the Dataverse repository 10.15783/C7J010. The Roman Dataset during the current study is available in the Dataverse repository 10.15783/C7QC7M. These datasets were derived from the following public domain resources: https://crawdad.org/epfl/mobility/20090224, https://crawdad.org/roma/taxi/20140717.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research is sponsored by the Science and Technology Planning Project of Sichuan Province under Grant No. 2020YFG0054, and the Joint Funds of the Ministry of Education of China. We would also like to thank JiangAn Chen for his comments, which helped improve part of this research.

References

[1] X. Niu, T. Chen, C. Q. Wu, J. Niu, and Y. Li, "Label-based trajectory clustering in complex road networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, 2020.

[2] Q. Yu, Y. Luo, C. Chen, and X. Zheng, "Road congestion detection based on trajectory stay-place clustering," *International Journal of Geo-Information*, vol. 8, no. 6, p. 264, 2019.

[3] H. Wang, Z. Xu, and S. Jia, "Cluster-indistinguishability: a practical differential privacy mechanism for trajectory clustering," *Intelligent Data Analysis*, vol. 21, no. 6, pp. 1305–1326, 2017.

[4] M. Ester, H. P. Kriegel, J. Sander, and X. Xu, *A density-based algorithm for discovering clusters in large spatial databases with noise*, pp. 226–231, 1996.

[5] A. Belhadi, Y. Djenouri, D. Djenouri, T. Michalak, and J. C. W. Lin, "Deep learning versus traditional solutions for group trajectory outliers," *IEEE Transactions on Cybernetics*, pp. 1–12, 2020.

[6] A. Belhadi, Y. Djenouri, G. Srivastava, A. Cano, and J. C. W. Lin, "Hybrid group anomaly detection for sequence data: application to trajectory data analytics," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–12, 2021.

[7] Y. Djenouri, D. Djenouri, and J. C. W. Lin, "Trajectory outlier detection," *ACM Transactions on Knowledge Discovery from Data*, vol. 15, no. 2, pp. 1–28, 2021.

[8] J. Gil Lee and J. Han, "Trajectory clustering: a partition-and-group framework," in *SIGMOD '07: Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, pp. 593–604, Beijing, China, 2007.

[9] B. Han, L. Liu, and E. R. Omiecinski, "A systematic approach to clustering whole trajectories of mobile objects in road networks," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 5, pp. 936–949, 2017.

[10] J. Queen, "Some methods for the classi_cation and analysis of multivariate observations," in *Proceedings of the Fifth Berkely Symposium on Mathematical Statistics and Probability*, Berkeley: University of California Press, 1966.

[11] Y. Yanagisawa, J. I. Akahani, and T. Satoh, *Shape-based similarity query for trajectory of mobile objects*, Springer, Berlin, Heidelberg, 2003.

[12] D. Berndt and J. Clifford, *Using dynamic time warping to find patterns in time series*, pp. 359–370, Advances in Knowledge Discovery and Data Mining, 1994.

[13] M. Vlachos, G. Kollios, and D. Gunopulos, "Discovering similar multidimensional trajectories," in *Proceedings 18th International Conference on Data Engineering*, San Jose, CA, USA, 2002.

[14] B. Lin and J. Su, "Shapes based trajectory for moving objects," in *13th ACM International Workshop on Geographic Information Systems, ACM-GIS 2005*, Bremen, Germany, November 2005.

[15] M. Gariel, A. N. Srivastava, and E. Feron, "Trajectory clustering and an application to airspace monitoring," *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, pp. 1511–1524, 2011.

[16] J. Qiu and R. Wang, "Inferring road maps from sparsely sampled gps traces," *Journal of Location Based Services*, vol. 10, pp. 111–124, 2016.

[17] C. Lu, Q. Sun, X. Ji, L. Xu, B. Wen, and M. Cheng, "A method of vehicle trajectory points clustering based on kernel distance," 2020.

[18] Y. U. Yan-Wei, J. Zhao-Fei, C. Lei, Z. Jin-Dong, L. Zhao-Wei, and L. Jing-Lei, "Fast density-based clustering algorithm for location big data," *Journal of Software*, vol. 29, no. 8, pp. 2470–2484, 2018.

[19] L. Zheng, D. Xia, X. Zhao et al., "Spatial-temporal travel pattern mining using massive taxi trajectory data," *Physica A-statistical Mechanics Its Applications*, vol. 501, pp. 24–41, 2018.

[20] Y. Mao, H. Zhong, H. Qi, P. Ping, and X. Li, "An adaptive trajectory clustering method based on grid and density in mobile pattern analysis," *Sensors*, vol. 17, no. 9, article 2013, 2017.

[21] B. Han, L. Liu, and E. Omiecinski, "Road-network aware trajectory clustering: integrating locality, Flow, and density," *IEEE Transactions on Mobile Computing*, vol. 14, no. 2, pp. 416–429, 2015.

[22] X. Niu, J. Zhu, C. Q. Wu, and S. Wang, "On a clustering-based mining approach for spatially and temporally integrated traffic sub-area division," *Engineering Applications of Artificial Intelligence*, vol. 96, article 103932, 2020.

[23] D. Kumar, H. Wu, S. Rajasegarar, C. Leckie, S. Krishnaswamy, and M. Palaniswami, "Fast and scalable big data trajectory clustering for understanding urban mobility," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 11, pp. 3709–3722, 2018.

[24] L. Bermingham and I. Lee, "A general methodology for *n*-dimensional trajectory clustering," *Expert Systems with Applications*, vol. 42, no. 21, pp. 7573–7581, 2015.

[25] A. Salarpour and H. Khotanlou, "Direction-based similarity measure to trajectory clustering," *IET Signal Processing*, vol. 13, no. 1, pp. 70–76, 2019.

[26] N. Guo, S. Shekhar, W. Xiong, L. Chen, and N. Jing, "Utsm: a trajectory similarity measure considering uncertainty based on an amended ellipse model," *International Journal of Geo-Information*, vol. 8, no. 11, p. 518, 2019.

[27] F. Liu and Z. Zhang, "Adaptive density trajectory cluster based on time and space distance," *Physica A: Statistical Mechanics and its Applications*, vol. 484, pp. 41–56, 2017.

[28] Q. Yu, Y. Luo, C. Chen, and S. Chen, "Trajectory similarity clustering based on multi-feature distance measurement," *Applied Intelligence*, vol. 49, no. 6, pp. 2315–2338, 2019.

[29] J. Domingo-Ferrer and R. Trujillo-Rasua, "Microaggregation- and permutation-based anonymization of movement data," *Information Sciences*, vol. 208, pp. 55–80, 2012.

[30] C. Wang, J. Yang, and J. P. Zhang, "Privacy preserving algorithm based on trajectory location and shape similarity," *Tongxin Xuebao/Journal on Communications*, vol. 36, 2015.

[31] J. Pan, Q. Jiang, and Z. Shao, "Trajectory clustering by sampling and density," *Marine Technology Society Journal*, vol. 48, no. 6, pp. 74–85, 2014.

[32] Y. Yang, J. Cai, H. Yang, J. Zhang, and X. Zhao, "Tad: a trajectory clustering algorithm based on spatial-temporal density analysis," *Expert Systems with Applications*, vol. 139, article 112846, 2020.

[33] W. Wang, F. Xia, H. Nie et al., "Vehicle trajectory clustering based on dynamic representation learning of internet of vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 6, pp. 3567–3576, 2021.

[34] A. Stefan, V. Athitsos, and G. Das, "The move-split-merge metric for time series," *IEEE Transactions on Knowledge Data Engineering*, vol. 25, no. 6, pp. 1425–1438, 2013.

[35] D. Yao, C. Zhang, Z. Zhu, J. Huang, and J. Bi, "Trajectory clustering via deep representation learning," in *2017 International Joint Conference on Neural Networks (IJCNN)*, Anchorage, AK, USA, 2017.

[36] N. R. Pal and J. Biswas, "Cluster validation using graph theoretic concepts," *Pattern Recognition*, vol. 30, no. 6, pp. 847–857, 1997.

[37] R. C. D. 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.

[38] L. Kun and Y. Jie, "Trajectory distance metric based on edit distance," *Journal of Shanghai Jiaotong University*, vol. 43, pp. 1725–1729, 2009.

[39] L. Chen, M. T. Ozsu, and V. Oria, "Robust and fast similarity search for moving object trajectories," in *SIGMOD '05: Proceedings of the 2005 ACM SIGMOD international conference on Management of data*, Baltimore, Maryland, USA, 2005.

Hindawi
Wireless Communications and Mobile Computing
Volume 2021, Article ID 7479473, 11 pages
https://doi.org/10.1155/2021/7479473

Research Article

Generating Bird's Eye View from Egocentric RGB Videos

Vanita Jain,[1] Qiming Wu,[2] Shivam Grover ,[1] Kshitij Sidana ,[1] Gopal Chaudhary,[1] San Hlaing Myint,[3] and Qiaozhi Hua [4]

[1]*Bharati Vidyapeeth's College of Engineering, New Delhi, India*
[2]*China Mobile (Hangzhou) Information Technologies Co., Ltd., Hangzhou, China*
[3]*Global Information and Telecommunication Institute, Waseda University, Tokyo, Japan*
[4]*Computer School, Hubei University of Arts and Science, Xiangyang, China*

Correspondence should be addressed to Qiaozhi Hua; 11722@hbuas.edu.cn

Received 3 August 2021; Revised 27 September 2021; Accepted 16 October 2021; Published 8 November 2021

Academic Editor: Jerry Chun-Wei Lin

In this paper, we present a method for generating bird's eye video from egocentric RGB videos. Working with egocentric views is tricky since such the view is highly warped and prone to occlusions. On the other hand, a bird's eye view has a consistent scaling in at least the two dimensions it shows. Moreover, most of the state-of-the-art systems for tasks such as path prediction are built for bird's eye views of the subjects. We present a deep learning-based approach that transfers the egocentric RGB images captured from a dashcam of a car to bird's eye view. This is a task of view translation, and we perform two experiments. The first one uses an image-to-image translation method, and the other uses a video-to-video translation. We compare the results of our work with homographic transformation, and our SSIM values are better by a margin of 77% and 14.4%, and the RMSE errors are lower by 40% and 14.6% for image-to-image translation and video-to-video translation, respectively. We also visually show the efficacy and limitations of each method with helpful insights for future research. Compared to previous works that use homography and LIDAR for 3D point clouds, our work is more generalizable and does not require any expensive equipment.

1. Introduction

Egocentric videos, commonly referred to as first-person videos, are captured from the POV of a subject (in our case from the POV of an autonomous vehicle). Egocentric videos are easy to capture and hence are accessible in real-time to the vehicle. However, they are deviously hard to for a computer to comprehend and work with. This is because egocentric videos are prone to occlusions, and there is a significant warping effect due to perspective which causes the objects closer to the camera to look inflated. Another drawback of the egocentric view is the nonlinear nature of objects in motion.

On the other hand, top-down views such as the views from a surveillance camera or drones show a more holistic and consistently scaled view of the environment, which makes them rich in data and easy to work with (see Figure 1). Previous work done in fields such as trajectory prediction is mainly focused on CCTV footage. State-of-the-art methods work irrespective of view but perform much better at top-down views of 45°or greater. Not only that, increasing the angle from 0° (eye level) to 90° (top-down) eliminates most of the occlusions and improves the visibility. With advancements in self-driving autonomous vehicle technology, it becomes important that we devise a way to overcome the shortcomings of egocentric perspective and make their accessibility useful [1–3].

In this paper, we present an approach for generating a bird's eye view of the environment from egocentric images. Unlike previous works [1, 4, 5] that use homography and/or perspective transform for estimating the coordinates of objects in a bird's eye view, we majorly aim to reconstruct the whole visible scene including the objects of interests (such as cars and pedestrians) and all other objects (such as buildings, trees, and crosswalks) that may affect the future behavior of the objects of interests. Our work is aimed at

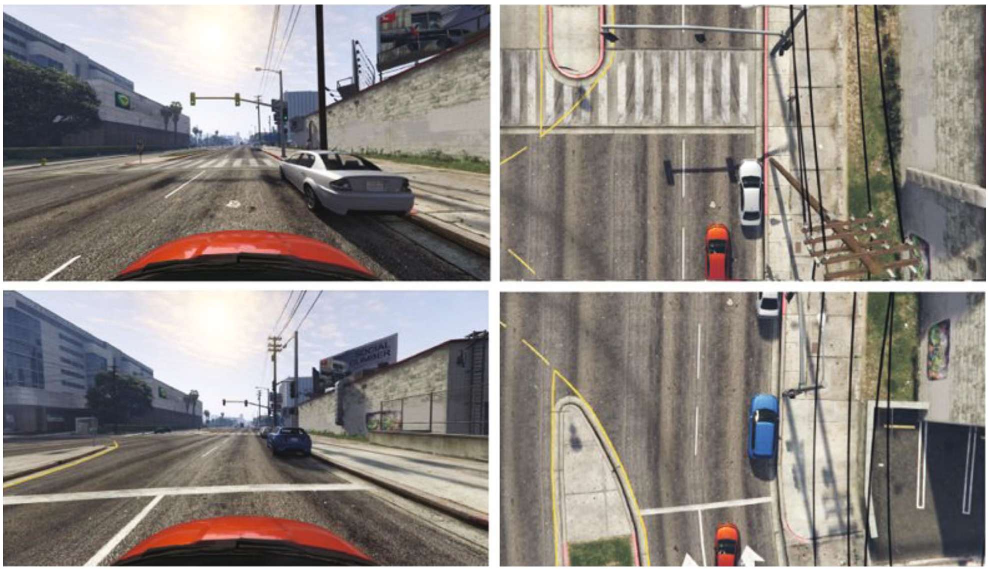


Figure 1: Example pairs of egocentric views (left row) and their corresponding bird's eye view (right row). The egocentric views are highly warped due to perspective and a major part of the environment is out of the field of camera's view. Bird's eye view shows a holistic view of the environment, and the scaling is consistent.

maintaining geometrical, spatial, and temporal consistency during the view translation. To the best of our knowledge, this has been an unexplored domain [6].

We identify this as a problem of view translation, and it can be solved through image-to-image or video-to-video translation, each having its own perks and shortcomings. We show experiments with both approaches and give directions for future research. We use an adversarial approach for the deep learning model takes as input an egocentric image and learns to generate its corresponding bird's eye view. Our work opens new avenues for progress in self-reliant and smarter autonomous systems [7, 8]. This also enables the development of smarter connected vehicles. Having egocentric views from multiple nearby vehicles, a much more accurate prediction of bird's eye view will be possible which is an area of interest for future research [9].

With the advent of Industry 5.0, interconnection of not only devices but also vehicles will be possible. Vehicles in proximity can collaboratively develop the novel viewpoint and fill in blind spot caused by occlusions [10]. Our work acts as a stepping stone towards making this possible [11].

2. Related Work

2.1. Classical Approach for View Translation. Perspective transformation is a classical approach to compensate for the camera angle. Using homography [4, 12–18], a plane is resolved, and the transformation is applied to correct the perspective. Since this approach relies on a mathematical approach to the problem, the resulting image can appear to be distorted and out of proportion.

2.2. 3D Point Cloud for View Translation. With the availability of technologies such as Lidar that readily give the 3D point clouds for the scene, obtaining a bird's eye representation for various applications [19–23] is relatively simple as compared to using RGB image as input. The LIDAR gives a readymade 3D point cloud of the environment which after some processing can be transformed into a 2D view from any specific angle. However, such sensors are expensive, and not all vehicles are equipped with them. Dashcams and cameras installed on mobile phones are generally incapable of inferring the 3D information and only provide RGB images. Our method uses a single RGB image as input, thus eliminating the use of any expensive equipment.

2.3. Learning-Based Approach. Learning-based approaches have been gaining popularity as they provide promising results in similar applications. This majorly includes those approaches in which we train our system to learn from a predisposed set of data. Convolutional neural networks have impacted the domain of image analysis greatly, and consequently, there have been works that use CNN along with other traditional methods such as homography to have a more dynamic approach towards generating bird's eye view from a single image. [18] uses a CNN to predict 4 parameters of the homography matrix which is used to transform the image into its bird's eye representation further. However, their model is majorly for images that already have some vertical leverage (for example from CCTV cameras) and would not be able to work on egocentric images such as those coming from a dashcam of a car, where the views are highly skewed with little scope for homography to work. In our work, we show an end-to-end approach for translating nonvertical egocentric images into their corresponding bird's eye views using a completely learning approach.

3. Methods

3.1. Dataset. We needed a dataset that has egocentric images (from a car's point of view) along with their corresponding bird's eye views. This poses three major constraints for bird's eye views (see Figure 2). (1) The pixel position of the subject car in all of bird's eye frames should be the same, in a way that the rest of the environment appears to be moving and the subject car appears to be stationary. (2) The camera angle in bird's eye view should also be such that a vertical line through the centre of the image should pass through vehicle's body perpendicularly. (3) The distance of the top-down camera from the car should also remain constant. A dataset satisfying all three of these requirements will allow for a consistent representation and avoid any discrepancy regarding the alignment and position of the camera during the image generation process.

Such a dataset is extremely hard to curate in the real world. Capturing the egocentric feed is easy and can be achieved by simply placing a fixed camera inside the car or on car's body. However, capturing bird's eye view is nearly impossible, especially with the constraints mentioned above. A plausible approach may be using a drone camera that hovers over the car. But keeping it stationary relative to the car is practically impossible.

So, we decided to make use of synthetic data for training purposes. Advances in graphics technology offer us hyperrealistic animation and games that we can make use of as an alternative for real-world data. One such game is Grand Theft Auto V (GTAV) in which the visuals of the environment and the behaviour of the cars and pedestrians mimic that of the real world. We make use of the SVA dataset released by [24] in which the camera changes between egocentric and bird's eye view at alternate time steps, which gives a highly accurate bird's eye representation for each egocentric frame. The camera also follows the constraints we mentioned above. Two sample sequences from the dataset can be seen in Figure 3. While the dataset released by [24] also contains bounding boxes, yaw, and other relevant information for nearby cars, we do not include that into our training process and leave that to future work.

3.2. View Translation. Before building a system, it is necessary to understand the data from the egocentric images that we would like to retain in bird's eye views. Taking the case of the view from a dashcam of a car, we not only want the objects of interest such as other cars and pedestrians to appear in bird's eye view but also the other aspects of the environment that may affect our or other cars and pedestrians' behaviours. For this, simply projecting the coordinates of the objects of interest in a top-down view is not enough. To this end, we treat this as a problem of view translation where we try to retain as much information as possible and describe how we achieve it below.

Image to image translation [25–35] is one such approach that generates images in one domain using images from another domain. This approach is best suited for isolated frames or images as it lacks temporal consistency. Video to video translation [36] is similar to image-to-image translation but improves upon temporal consistency [37]. We talk about how we made use of these in our work and how well they perform compared to each other.

The major task at this point is to generate a bird's eye view y given an egocentric input x. Generative adversarial networks (GANs) [38] have performed remarkably well in the deep learning-based generative area of study. The architecture of a GAN consists of two parts: a generator G and a discriminator D. The generator is supposed to generate unseen but realistic data that falls in a similar domain as the training dataset, and the job of the discriminator is to classify a generated data point as realistic or fake. G and D are both trained together in a two-player min-max situation, where we try to establish a Nash equilibrium. But simple GANs are only effective in generative image synthesis applications if we need to generate new examples of images. We basically have no control over the data being generated [39]. To be able to control the outputs and to make use of additional information, such as class labels, or in our case, an input image of egocentric domain x that we want to be translated into an image of bird's eye domain y, we use an extension of GANs called conditional GANs [40, 41].

In conditional generative adversarial networks, the generator G learns to generate fake samples with a conditioned data point of domain x instead of unknown noise distribution as in simple GANs. The final objective of a conditional GAN looks like the following:

$$\mathscr{L}_{\mathrm{cGAN}}(G, D) = \mathbb{E}_{x,y}[\log D(x, y)] + \mathbb{E}_{x,z}[\log (1 - D(x, y))]. \tag{1}$$

In the task of image-to-image translation, the condition input is an image of domain x, and the generator outputs its corresponding image in the search space of domain y. There has been quite some progress in the field of image-to-image translation when combined with conditional GANs. Conditional GANs for image-to-image translation has been used to achieve tasks like colourization of black and white images by Zhang et al. [42], future frame prediction [43, 44], and image prediction from normal maps [45, 46]. We build on the work by Isola et al. [26] which consists of a general image to image translation network. They also incorporate a convolutional PatchGAN classifier for the discriminator which allows the structure to penalize at the scale of image patches. So, instead of trying to check whether the image as a whole is real or not, the PatchGAN checks whether each $N \times N$ patch in the image fed to the discriminator is real or not [47]. Then, the predictions by the discriminator for all patches are averaged and given out as the final output.

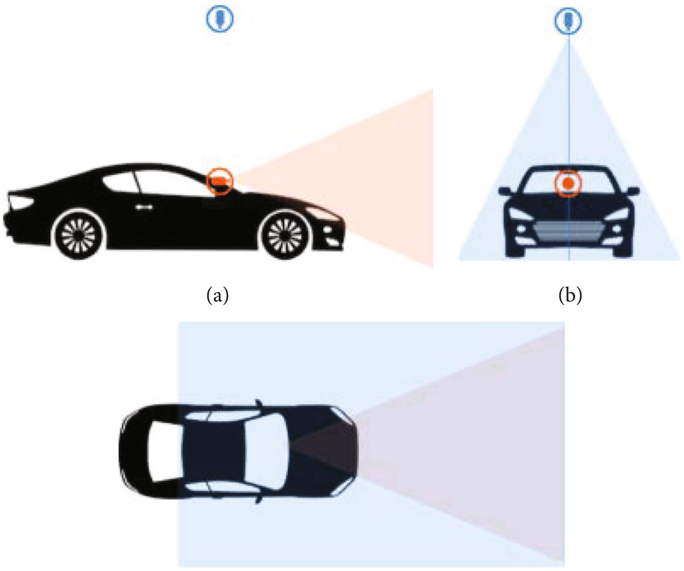
Along with the cGAN loss in Equation (1), they also use a traditional L1 loss. This forces the generator to generate images near the ground truth output in an L1 sense while also trying to fool the discriminator into believing the generated images are real.

$$\mathscr{L}_{L1}(G) = \mathbb{E}_{x,y,z}[\|y - G(x, y)L1\|. \tag{2}$$

This results in their final objective function as

$$G^* = \arg \min_G \max_D \mathscr{L}_{\mathrm{cGAN}}(G, D) + \lambda \mathscr{L}_{L1}(G). \tag{3}$$

Apart from the PatchGAN, their generator network uses



(a)

(b)

(c)

Figure 2: An overview of the camera placements for collecting the dataset. The camera with orange field of view captures the egocentric view, and the camera with blue field of view captures bird's eye view. The position (constant) and a range of the camera views (not to scale) are demonstrated as well.

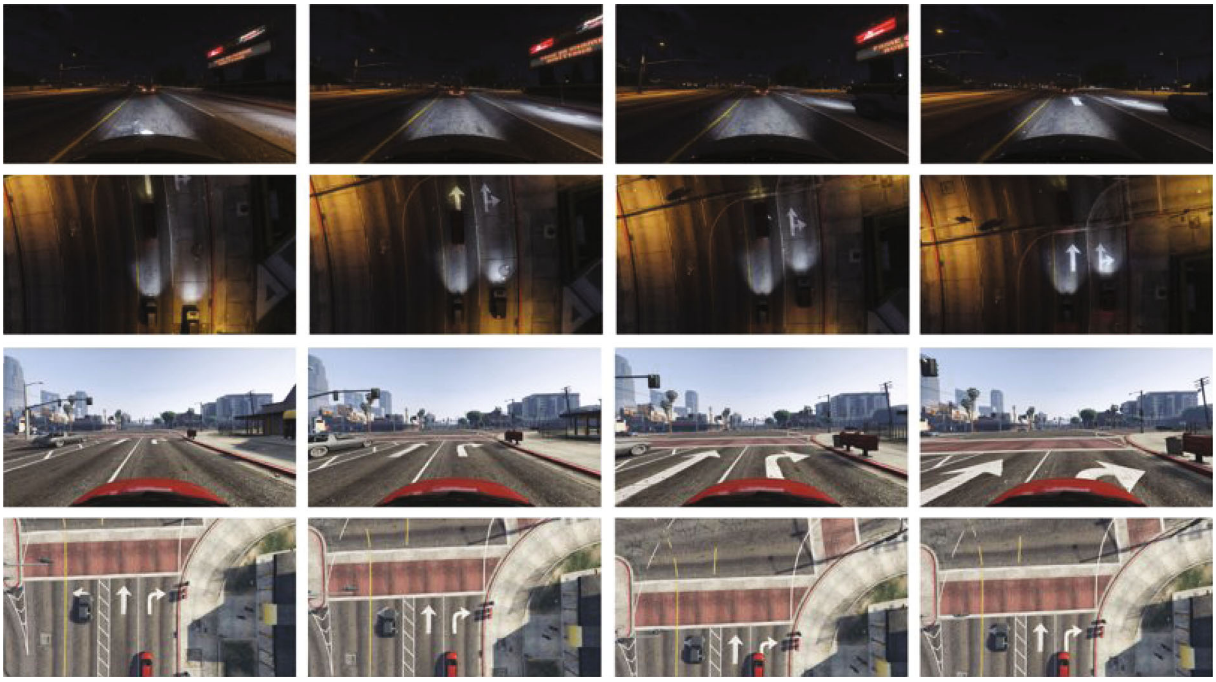


Figure 3: Sample images from the dataset we used. Each pair of rows are a different sequence, and the top row in each sequence shows the egocentric views, and the bottom shows bird's eye views.

a U-Net [48] style architecture which allows them to establish a better relationship between the input and output images that have the same low-level structure such as in image colourization and simulation to reality. In our case, this feature is not as useful since our input images and output images are considerably different, and this does not prove to be a disadvantage either as the network without the U-Net architecture gave similar results to the original Pix2Pix network. We also show the quantitative comparison of both in the next section.

For each step in training the network, we randomly pick an egocentric image from the sequences and give it as the input along with its corresponding bird's eye view as the ground truth label. An overview of the training process can

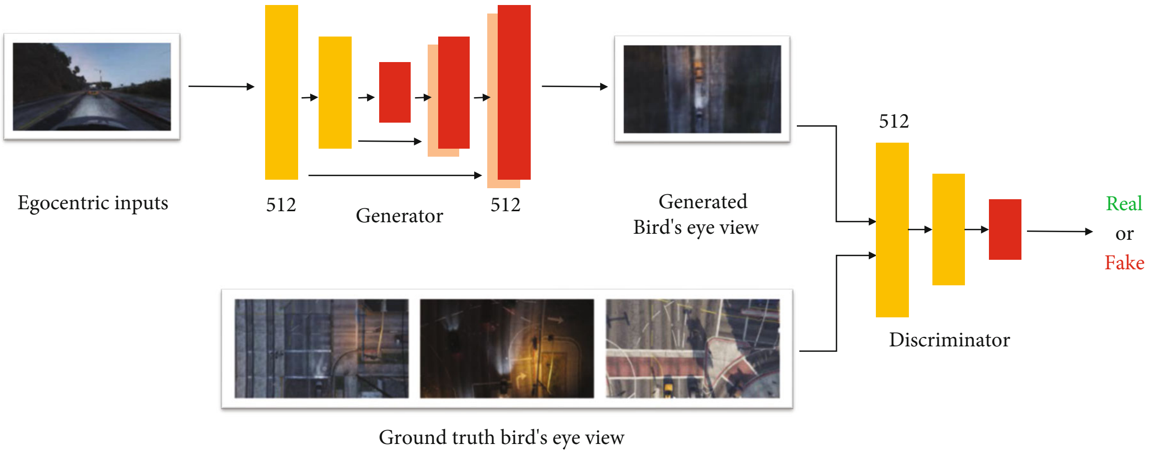


Figure 4: Training pipeline for image-to-image translation. The egocentric image is sent into the generator, and the generator outputs a predicted bird's eye view, which is compared to the ground truth view (not shown). To make the results look realistic, a discriminator is also trained simultaneously that predicts whether the generated image is real or not.

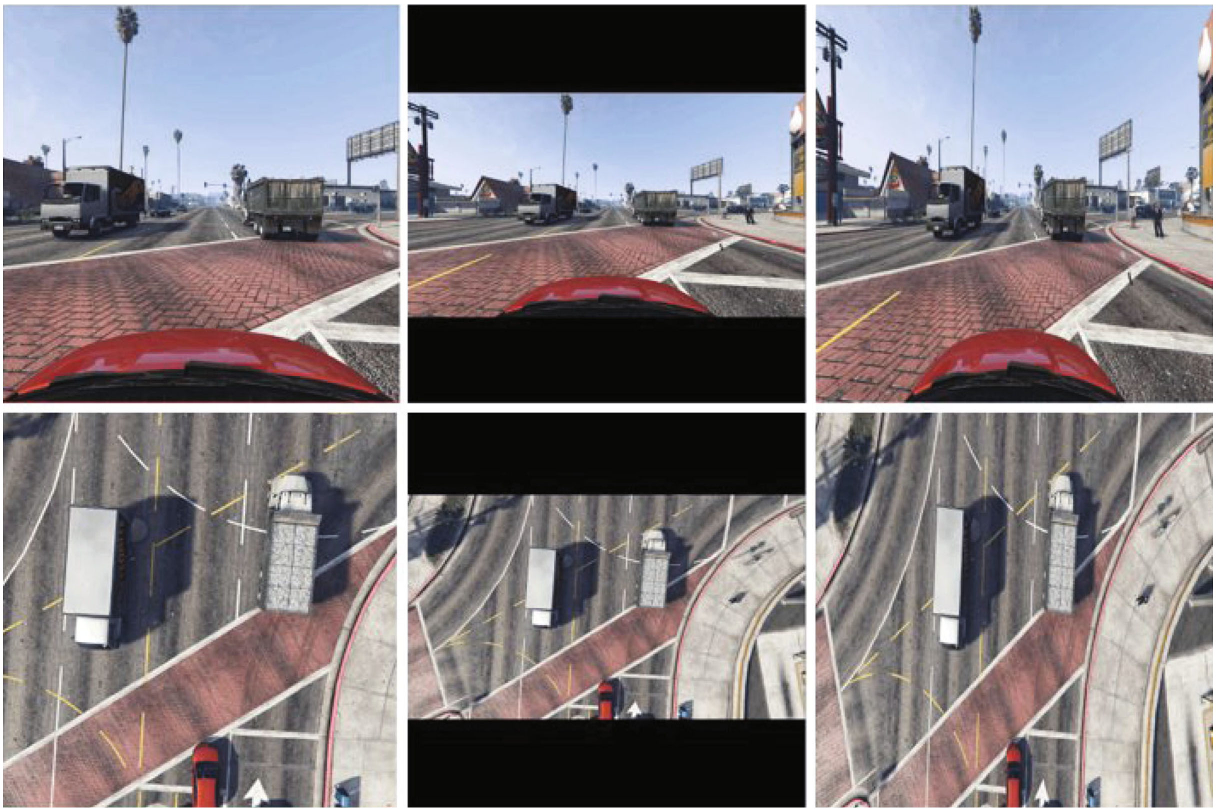


Figure 5: Each column represents three different methods for providing the egocentric (top) and bird's eye view (bottom) images to the image translation model.

be seen in Figure 4 The images were originally of the aspect ratio 16 : 9, whereas the network takes as input images with a 1 : 1 aspect ratio (or square images). To solve this, we could do three things as shown in Figure 5. (1) Centre crop the image as a square. This however leaves out the peripheral vision which is very important especially for autonomous vehicles since it is important to keep track of the vehicles that are trying to overtake you. (2) Add padding to the top and bottom of the original image so that it turns into a square. Unlike centre cropping, this does not leave out any information present in the original image. However, the issue with this was that almost a quarter of the image space was being wasted on padding. (3) Resize the image into a square. In this approach, no information is being left out, and the space is being utilised efficiently. The only issue with this method is that the internal aspect ratio of the original image is ruined which makes the image look squished. However, this does not seem to affect the learning of the model

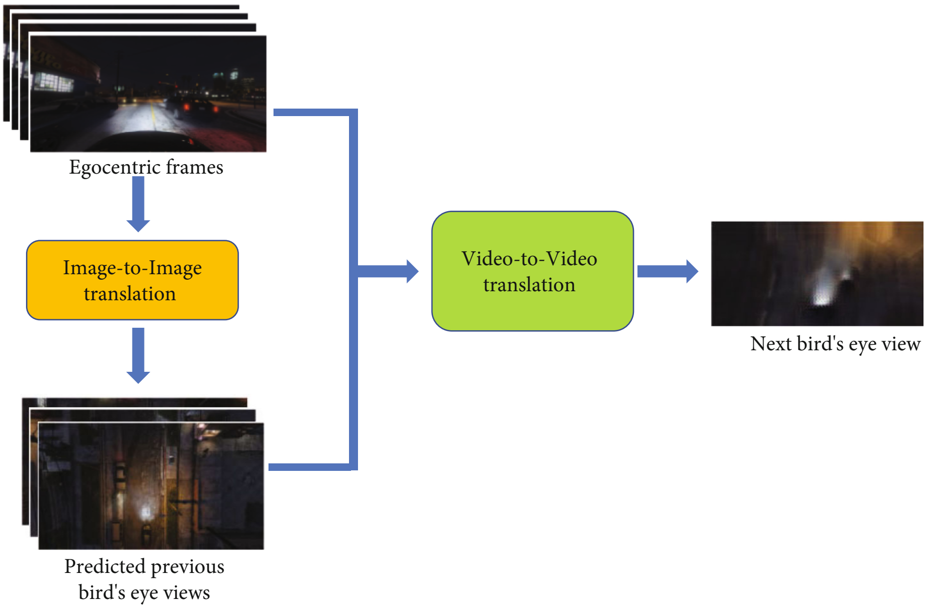

Egocentric frames
Image-to-Image translation
Predicted previous bird's eye views
Video-to-Video translation
Next bird's eye view

Figure 6: Training pipeline for video-to-video translation. For the first three egocentric frames, we use the image-to-image translation module to generate rough predictions of bird's eye view. All these along with the fourth egocentric views are input to the model, and the model generates bird's eye view for the fourth frame. Then, for the fifth frame, we also send the previously generated output as the label for the fourth frame, and this goes on until all frames have been processed.

negatively and seemed to be the most effective out of all three methods. Further efforts on evaluating the three different methods can be seen in the experiments section.

Since the application of our work is primarily in a video-based task, we also decided to use a temporally consistent model [36] for training. In this approach, the model requires us to send a sequence of frames as input instead of a single frame as we saw in image-to-image translation [26]. The model works in a coarse-to-fine way, i.e., first, a low-resolution model takes as input an image and along with a sequence of previous output images. For the very first image, we use the image-to-image translation model for generating the previous output images. Then, the generator outputs the next frame (Figure 6). Then, higher resolution generator is stacked on top of this generator which is used to increase the resolution of the generated frame. Once the model starts to predict the next frame in the sequence, we then use its predicted frame for subsequent inputs (this also deteriorates the input quality for the next frame, which might cause a significant cascading effect and the quality of the predictions decrease continuously) (Figure 7). We use images of sizes 1024 × 512 which require us to use two generators. The first one outputs images of size 512 × 256, and the second one gives us the final output.

4. Results

In this section, we will show and evaluate the results of our view translation pipeline.

4.1. Image-to-Image Translation. For image-to-image translation, the first experiment that we conducted was to establish the best method to crop and resize the images before feeding them into the model as ground truth. We checked three different methods of arranging them as seen in Figure 5. We trained a model three times on the same dataset and each time preprocessing the images differently. We did a qualitative and quantitative analysis for establishing which method is the best. For qualitative analysis, we did a user study with 5 human subjects and asked them to rate the generated images from each method on a scale of 1 to 10 on three factors: image quality, amount of crucial information retained, and the amount of details in the image (Table 1).

Note that for the third method, we resized the images into a square and sent that into the model. The generated image was a square as well. However, since it looked squished, we inverted the resize factor of the generated image back to their original aspect ratio so that they look natural to the users.

To quantitatively evaluate the different methods, we checked the mean structural similarity index and the root mean square error between the output images and their corresponding ground truth images on a test dataset containing 20 images. We show the average values in Table 2 where method I correspond to Figure 5 (first column), method II corresponds to Figure 5 (second column), and method III corresponds to Figure 5 (third column). After testing, it seemed intuitive to use method III for the final training.

Next, we show the results of the final image-to-image translation model on unseen input images in Figure 8. On comparing the generated results with the ground truth, we get the average SSIM value as 0.72 and RMSE value as 30.56. We also tested the model with the U-net with skip connections, and we got nearly the same results with an average SSIM value of 0.712 and RMSE value of 28.25. To quantitatively evaluate the details, present in the generated

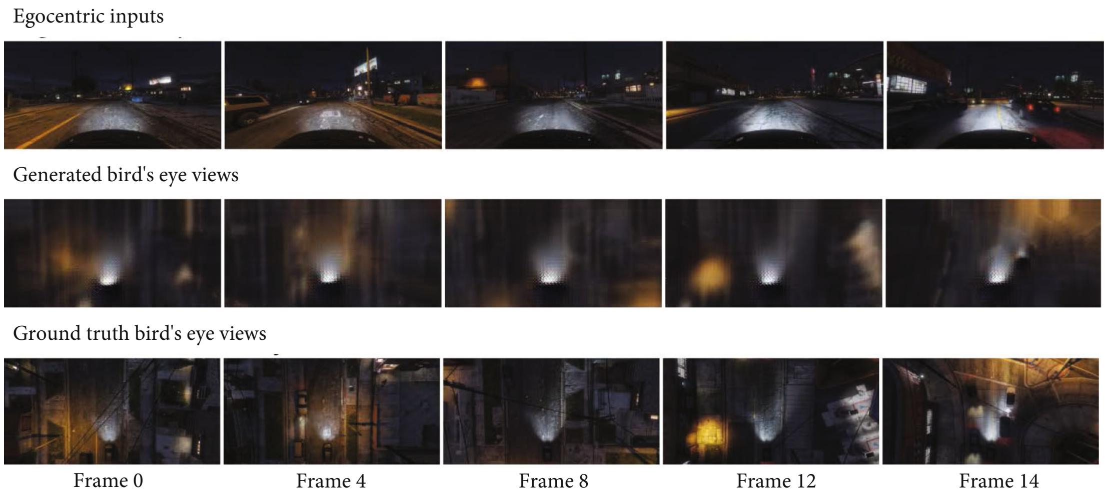


Figure 7: Results of the video-to-video translation experiments on a test sequence.

Table 1: A user study on different types of generated images from differences in preprocessing.

Factor	I	II	III
Image quality	8.5	6	8.5
Crucial information present	6.5	8	8.5
Details persevered	8.5	5	8

Table 2: Quantitative analysis of different types of generated images from differences in preprocessing.

	I	II	III
RMSE	30.1	35.4	32.19
SSIM	0.62	0.51	0.65

images, we further perform edge detection using a Canny edge detector on multiple predicted images and their corresponding ground truth images.

On comparing the ground truth edges with the edges in the generated images, we get an average SSIM score of 0.761 and an average RMSE score of 70.54. With the skip connections, we got an average SSIM score of 0.728 and an average RMSE score of 68.25. In Figure 9, we show three good results (retained most of the useful details, such as shapes of cars and crosswalks) bounded with a green and three failure cases (did not retain much useful details) bounded with a red box. The model is even able to understand subtle details such as the headlights being on in the vehicles. On a visual observation of the generated images, the results seem blurry and do not quite capture the environment exactly as in the ground truth images. This is a limitation of the type of model we selected, and we talk more about this in the discussion section and also mention the research areas that might help in tackling this issue. In Figure 10, we also compare our results with the results obtained from homographic transformations. We compare the nonblank parts of the image with the corresponding parts in the ground truth image and get an average SSIM of 0.41 and an average RMSE of 47.0. Compared to the homographic results, our image-to-image translation results are better by a margin of 77% for SSIM and 40% for RMSE. Visually, the homographic results look very distorted, and the objects cannot be reliably detected.

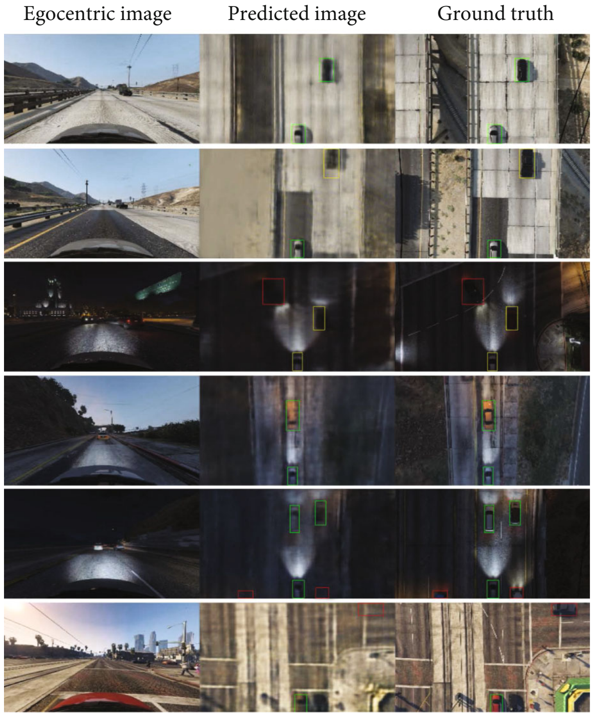


Figure 8: Sample results of the image-to-image translation method. Bounding boxes are given in second and third columns to show the actual positions of vehicles. A green bounding box signifies a successful reconstruction including the position of the reconstructed vehicle, whereas a red bounding box signifies an unsuccessful or missing reconstruction.

We finally conducted experiments for video-to-video translation. In Figure 7, we show the results for a test sequence of 14 frames. The model is able to reconstruct bird’s eye view and successfully captures details such as

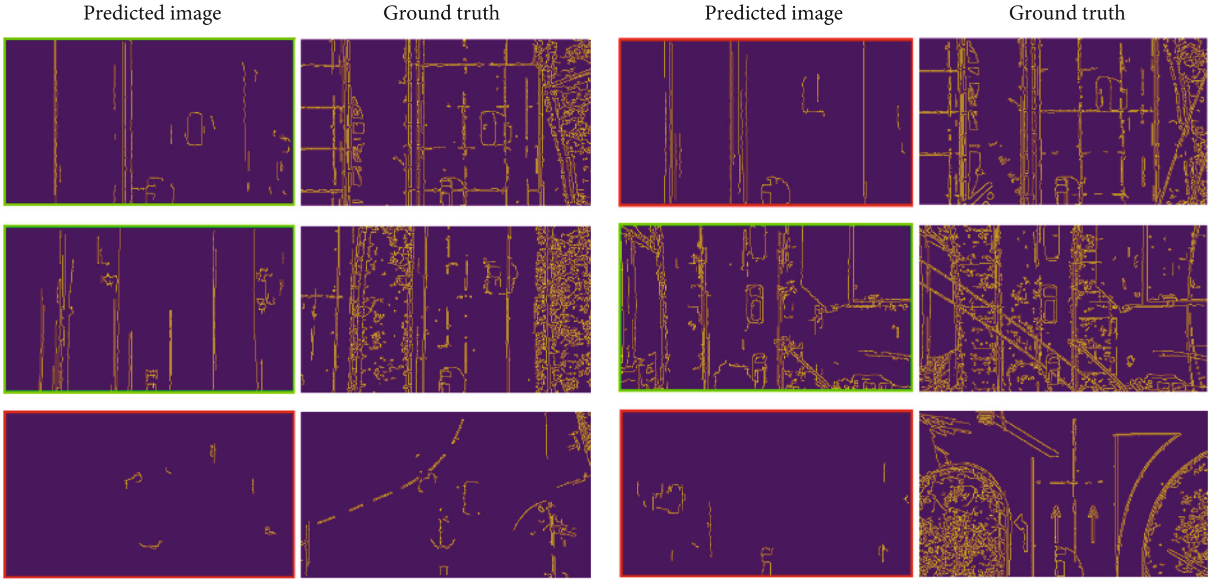


Figure 9: Comparison of detected edges on predicted and ground truth bird's eye view images. The generated images outlined with green retain the useful details such as shapes of cars, roads, and buildings, whereas the generated images outlined with red fail to retain useful details.

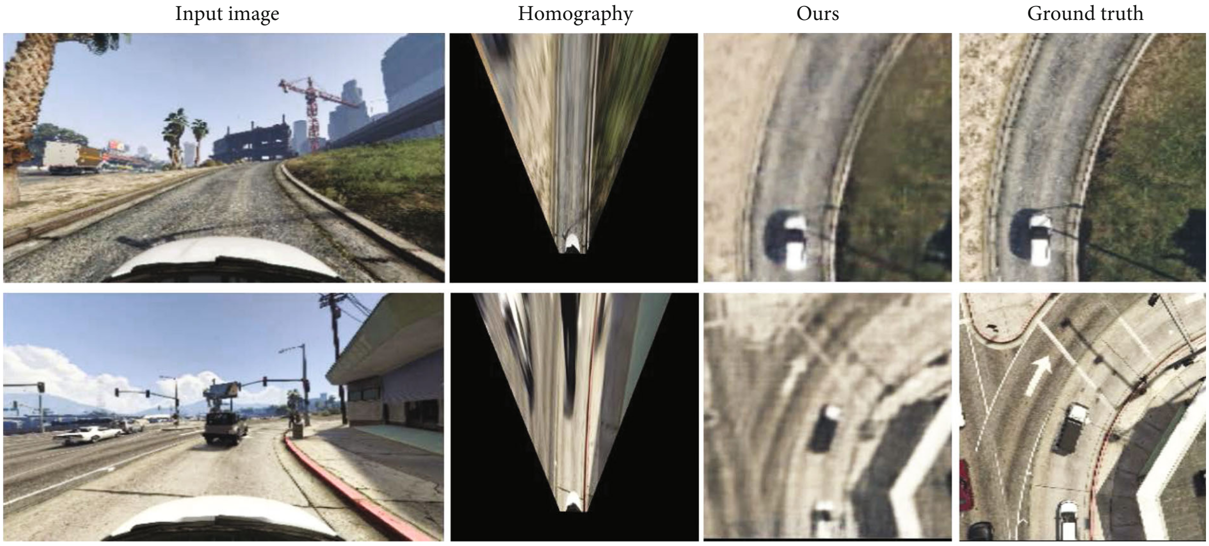


Figure 10: Comparison of our method with homography.

nearby cars, headlights, and ambient lights in a temporally consistent manner. On the negative side, the results are blurry. We talk about why this is so in the discussions section below and also mention the possible solutions. Initial results are better than the future frames, and the details start to deteriorate as more frames are predicted by the model. This happens because, for each consequent step, the model takes as input the previously generated frame, which propagates the errors forward deteriorating the quality of each consequent image. To evaluate the results quantitatively, we compared the generated bird's eye views and their corresponding ground truths and got the calculated RMSE value as 40.25 and the SSIM as 0.47. Compared to the homographic results, our video-to-video translation results are better by a margin of 14.4% for SSIM and 14.6% for RMSE.

We also show the comparison between the two methods in Figure 11. We ran both models on the same set of 6 frames of multiple sequences. In Figure 11(a), we show the abilities of the model to generate images that are similar to the actual ground truth. For this, we simply calculate the SSIM values of each generated image and its corresponding ground-truth bird's eye view. The SSIM values for image-to-image translation do not follow any trend; however, the values for video-to-video translation degrade as more frames are generated. This is due to the cascading effect on errors in each generated frame being propagated forward. In

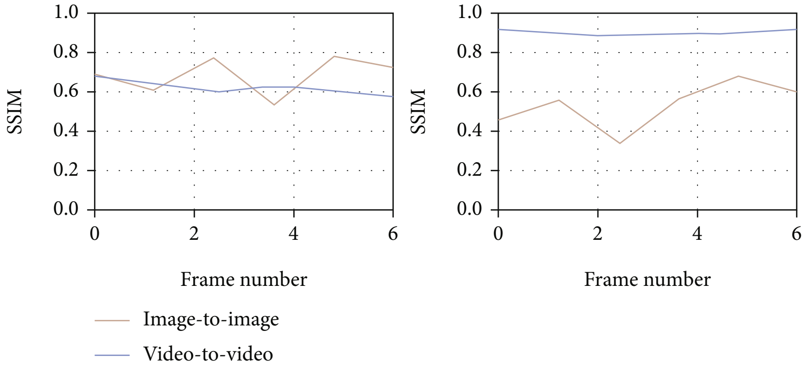


FIGURE 11: Comparison of the results from image-to-image and video-to-video translation methods. (a) The SSIM values of each generated frame with its corresponding ground-truth frame. The SSIM values in (a) for the image-to-image method do not seem to follow any trend, whereas for the video-to-video translation method, the quality of the image seems to degrade a little as more frames are generated. (b) The SSIM values of each generated frame with its previous generated frame. In (b), the consecutive frames from image-to-image translation show little similarity, whereas the consecutive frames from video-to-video translation show high similarity and hence consistency.

Figure 11(b), we compare the consistency and similarity in the consecutive frames generated from both methods. For this, we find the SSIM between a generated frame and the frame generated before it. It should be noted that even in the most ideal case, the value will never be 1 as the temporal change in the egocentric images will always incur a change in bird's eye view. However, a high value still shows that there is a good level of consistency in the consecutive frames. Video-to-video translation shows high levels of consistency, whereas image-to-image translation gives low SSIM values.

5. Discussions and Future Works

Our work shows the possibility of using RGB egocentric images for inferring bird's eye view around the subject vehicle. The failure results of work also provide key insights and directions that may benefit future researchers. Architectures such as [26, 36] work better for translations that have some level of geometric alignment, for example, horse-to-zebra or oranges-to-apples, where the input image and the output image are geometrically and structurally very similar, with differences only in the appearances and textures. However, in the task that we aimed to solve, there is a high level of geometric deformation in the input and output images. Egocentric images are completely different from top-down images, and even though this difference is consistent in all such images, models such as [26, 36] are not well-equipped for this. In order to solve the issue of geometric deformation in such images, future works may look at deformable convolutional networks [49], proposed by Dai et al., and deformable skip-connections [50], proposed by Siarohin et al. Since the motivation for this work came from the expensiveness of sensors such as Lidar, we discourage the use of such sensors. However, using deep learning methods for estimating depth data is also an area of interest for future work.

6. Conclusion

In this paper, we presented an end-to-end method for translating egocentric views from RGB cameras such as those installed on vehicles into bird's eye views of the environment the subject vehicle was present in. One of the biggest hurdles is that egocentric views have a high level of distortion due to perspective, whereas a bird's eye view has a consistent scaling. The two are quite opposite in terms of geometric alignment. Previous traditional methods such as handcrafted homography transformations are not generalizable, and they do not work very well for views with minimal vertical leverage (e.g., view from the dashcam). More modern methods that use external sensors such as LIDAR can be very costly and computationally extensive. Taking all this into consideration, we develop our method to only use RGB frames from a single inexpensive camera installed in the car and so that it can be used for inference on the go on most modern mobile systems. We treat this as a task of view translation and implement it for two different use cases, one where we have a single image and one where we have a sequence of frames. We use an adversarial approach for training the model and experiment with image-to-image and video-to-video translation. The results from both experiments show that this can be a reliable approach to perform this task, and in the future, it can be used in the real world. However, there do exist some limitations, such as artefacts and loss of details over time, and we provide key insights for future researchers on how the performance and accuracy can be improved for this specific task. The work opens up new avenues for research on environment sensing in autonomous vehicles that only use dashcams as a sensor. While we have only shown the efficacy of this work for vehicle data, this can be extended to all sorts of egocentric views such as wearable cameras, and cameras installed on domestic assistant robots.

Data Availability

All data generated or analyzed during this study are included in this published article. Data is available at https://aimagelab.ing.unimore.it/imagelab/page.asp?IdPage=19.

Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

References

[1] Y. Jiang, F. Gao, and G. Xu, "Computer vision-based multiple-lane detection on straight road and in a curve," in *2010 International Conference on Image Analysis and Signal Processing*, pp. 114–117, Zhejiang, China, April 2010.

[2] K. Yu, L. Tan, S. Mumtaz et al., "Securing critical infrastructures: deep learning-based threat detection in the IIoT," *IEEE Communications Magazine*, 2021.

[3] K. Yu, Z. Guo, Y. Shen, W. Wang, J. C. Lin, and T. Sato, "Secure artificial intelligence of things for implicit group recommendations," *IEEE Internet of Things Journal*, 2021.

[4] A. Agarwal, C. V. Jawahar, and P. J. Narayanan, *A Survey of Planar Homography Estimation Techniques*, Centre for Visual Information Technology, Tech. Rep. IIIT/TR/2005/12, 2005.

[5] K. Yu, M. Arifuzzaman, Z. Wen, D. Zhang, and T. Sato, "A key management scheme for secure communications of information centric advanced metering infrastructure in smart grid," *IEEE Transactions on Instrumentation and Measurement*, vol. 64, no. 8, pp. 2072–2085, 2015.

[6] L. Tan, H. Xiao, K. Yu, M. Aloqaily, and Y. Jararweh, "A blockchain-empowered crowdsourcing system for 5G-enabled smart cities," *Computer Standards Interfaces*, vol. 76, p. 103517, 2021.

[7] L. Tan, K. Yu, A. K. Bashir et al., "Toward real-time and efficient cardiovascular monitoring for COVID-19 patients by 5G-enabled wearable medical devices: a deep learning approach," *Neural Computing and Applications*, 2021.

[8] L. Tan, K. Yu, F. Ming, X. Chen, and G. Srivastava, "Secure and resilient artificial intelligence of things: a HoneyNet approach for threat detection and situational awareness," *IEEE Consumer Electronics Magazine*, p. 1, 2021.

[9] L. Tan, N. Shi, K. Yu, M. Aloqaily, and Y. Jararweh, "A blockchain-empowered access control framework for smart devices in green internet of things," *ACM Transactions on Internet Technology*, vol. 21, no. 3, pp. 1–20, 2021.

[10] V. Scinteie, *Autonomous vehicles and industry 5.0 are the basis for future-oriented traffic concepts in smart cities*, 2019.

[11] C. Feng, K. Yu, A. K. Bashir et al., "Efficient and secure data sharing for 5G flying drones: a blockchain-enabled approach," *IEEE Network*, vol. 35, no. 1, pp. 130–137, 2021.

[12] P. Jain and C. Jawahar, "Homography Estimation from Planar Contours," *IEEE*, pp. 877–884, 2006.

[13] X. Li, X. Fang, C. Wang, and W. Zhang, "Lane detection and tracking using a parallel-snake approach," *Journal of Intelligent Robotic Systems*, vol. 77, no. 3-4, pp. 597–609, 2015.

[14] I. S. Kholopov, "Bird's eye view transformation technique in photogrammetric problem of object size measuring at low-altitude photography," *Advances in Engineering Research*, vol. 133, 2017.

[15] A. Abbas and A. Zisserman, "A geometric approach to obtain a bird's eye view from an image," in *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pp. 4095–4104, Seoul, Korea (South), October 2019.

[16] H. Li, K. Yu, B. Liu, C. Feng, Z. Qin, and G. Srivastava, "An efficient ciphertext-policy weighted attribute-based encryption for the Internet of health things," *IEEE Journal of Biomedical and Health Informatics*, p. 1, 2021.

[17] H. Le, F. Liu, S. Zhang, and A. Agarwala, "Deep homography estimation for dynamic scenes," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7652–7661, Seattle, WA, USA, June 2020.

[18] S. A. Abbas and A. Zisserman, "A geometric approach to obtain a bird's eye view from an image," in *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, Seoul, Korea (South), October 2020.

[19] J. Zarzar, S. Giancola, and B. Ghanem, "Efficient bird eye view proposals for 3D Siamese tracking," 2019, http://arxiv.org/abs/1903.10168.

[20] A. E. Sallab, I. Sobh, M. Zahran, and N. Essam, "Lidar sensor modeling and data augmentation with GANs for autonomous driving," 2019, http://arxiv.org/abs/1905.07290.

[21] J. Zarzar, S. Giancola, and B. Ghanem, "Efficient tracking proposals using 2D-3d Siamese networks on Lidar," 2019, http://arxiv.org/abs/1903.10168.

[22] L. Zhen, A. K. Bashir, K. Yu, Y. D. al-Otaibi, C. H. Foh, and P. Xiao, "Energy-efficient random access for LEO satellite-assisted 6G Internet of remote things," *IEEE Internet of Things Journal*, vol. 8, no. 7, pp. 5114–5128, 2021.

[23] L. Zhen, Y. Zhang, K. Yu, N. Kumar, A. Barnawi, and Y. Xie, "Early collision detection for massive random access in satellite-based Internet of Things," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 5, pp. 5184–5189, 2021.

[24] A. Palazzi, G. Borghi, D. Abati, S. Calderara, and R. Cucchiara, "Learning to map vehicles into bird's eye view," in *International Conference on Image Analysis and Processing*, pp. 233–243, Springer, 2017.

[25] K. Yu, L. Tan, L. Lin, X. Cheng, Z. Yi, and T. Sato, "Deep-learning-empowered breast cancer auxiliary diagnosis for 5GB remote E-health," *IEEE Wireless Communications*, vol. 28, no. 3, pp. 54–61, 2021.

[26] P. Isola, J.-Y. Zhu, T. Zhou, and A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5967–5976, Honolulu, 2017.

[27] X. Huang, M.-Y. Liu, S. Belongie, and J. Kautz, "Multimodal unsupervised image-to-image translation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 172–189, Munich, Germany, 2018.

[28] K. Bousmalis, N. Silberman, D. Dohan, D. Erhan, and D. Krishnan, "Unsupervised pixel-level domain adaptation with generative adversarial networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 95–104, Honolulu, 2017.

[29] M.-Y. Liu, T. Breuel, and J. Kautz, "Unsupervised image-to-image translation networks," in *31st Conference on Neural Information Processing Systems (NIPS 2017)*, Long Beach, CA, USA, 2017.

[30] M. Y. Liu and O. Tuzel, "Coupled generative adversarial networks," *Advances in neural information processing systems*, vol. 29, pp. 469–477, 2016.

[31] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb, "Learning from simulated and unsupervised images through adversarial training," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2107–2116, Hololulu, 2017.

[32] Y. Taigman, A. Polyak, and L. Wolf, "Unsupervised cross-domain image generation," *arXiv preprint arXiv*, vol. 1611, Article ID 02200, 2016.

[33] T. C. Wang, M. Y. Liu, J. Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro, "High-resolution image synthesis and semantic manipulation with conditional gans," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 8798–8807, 2018.

[34] J.-Y. Zhu, T. Park, P. Isola, and A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of the IEEE international conference on computer vision*, pp. 2223–2232, 2017.

[35] J.-Y. Zhu, R. Zhang, D. Pathak et al., *Toward Multimodal Image-to-Image Translation*, 2017.

[36] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu et al., "Video-to-video synthesis," *arXiv preprint arXiv*, vol. 1808, Article ID 06601, 2018.

[37] Z. Guo, Y. Shen, A. K. Bashir, K. Yu, and J. C. W. Lin, "Graph embedding-based intelligent industrial decision for complex sewage treatment processes," *International Journal of Intelligent Systems*, 2021.

[38] I. Goodfellow, J. Pouget-Abadie, M. Mirza et al., "Generative adversarial nets," in *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds., pp. 2672–2680, Curran Associates, Inc., 2014.

[39] K. Yu, L. Lin, M. Alazab, L. Tan, and B. Gu, "Deep learning-based traffic safety solution for a mixture of autonomous and manual vehicles in a 5G-enabled intelligent transportation system," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 7, pp. 4337–4347, 2021.

[40] Z. Guo, K. Yu, A. Jolfaei, A. K. Bashir, A. O. Almagrabi, and N. Kumar, "A fuzzy detection system for rumors through explainable adaptive learning," *IEEE Transactions on Fuzzy Systems*, p. 1, 2021.

[41] M. Mirza and S. Osindero, "Conditional generative adversarial nets," 2014, https://arxiv.org/abs/1411.1784.

[42] R. Zhang, P. Isola, and A. A. Efros, "Colorful image colorization," *ECCV*, 2016.

[43] Z. Guo, K. Yu, Y. Li, G. Srivastava, and J. C.-W. Lin, "Deep learning-embedded social Internet of Things for ambiguity-aware social recommendations," *IEEE Transactions on Network Science and Engineering*, 2021.

[44] M. Mathieu, C. Couprie, and Y. Lecun, *Deep Multi-Scale Video Prediction Beyond Mean Square Error*, 2015.

[45] Z. Guo, L. Tang, T. Guo, K. Yu, M. Alazab, and A. Shalaginov, "Deep graph neural network-based spammer detection under the perspective of heterogeneous cyberspace," *Future Generation Computer Systems*, vol. 117, pp. 205–218, 2021.

[46] X. Wang and A. Gupta, "Generative image modeling using style and structure adversarial networks," in *Computer Vision – ECCV 2016*, vol. 9908, pp. 318–335, Springer, 2016.

[47] K. Yu, L. Tan, M. Aloqaily, H. Yang, and Y. Jararweh, "Blockchain-enhanced data sharing with traceable and direct revocation in IIoT," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 11, pp. 7669–7678, 2021.

[48] O. Ronneberger, P. Fischer, and T. Brox, "U-net: convolutional networks for biomedical image segmentation," 2015, https://arxiv.org/abs/1505.04597.

[49] J. Dai, H. Qi, Y. Xiong et al., "Deformable convolutional networks," in *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 764–773, Venice, Italy, October 2017.

[50] A. Siarohin, E. Sangineto, S. Lathuilière, and N. Sebe, "Deformable GANs for pose-based human image generation," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3408–3416, Salt Lake City, UT, USA, June 2018.

Hindawi
Wireless Communications and Mobile Computing
Volume 2021, Article ID 1230279, 12 pages
https://doi.org/10.1155/2021/1230279

Research Article

Object-Level Remote Sensing Image Augmentation Using U-Net-Based Generative Adversarial Networks

Jian Huang,[1] **Shanhui Liu**,[2] **Yutian Tang**,[3] **and Xiushan Zhang**[1]

[1]*School of Electrical Engineering, Naval University of Engineering, Hubei, China*
[2]*School of Information Science and Technology, Hubei University, Hubei, China*
[3]*School of Information Science and Technology, ShanghaiTech University, Shanghai, China*

Correspondence should be addressed to Yutian Tang; tangyt1@shanghaitech.edu.cn

Received 25 July 2021; Accepted 24 August 2021; Published 9 September 2021

Academic Editor: Yulin Wang

With the continuous development of deep learning in computer vision, semantic segmentation technology is constantly employed for processing remote sensing images. For instance, it is a key technology to automatically mark important objects such as ships or port land from port area remote sensing images. However, the existing supervised semantic segmentation model based on deep learning requires a large number of training samples. Otherwise, it will not be able to correctly learn the characteristics of the target objects, which results in the poor performance or even failure of semantic segmentation task. Since the target objects such as ships may move from time to time, it is nontrivial to collect enough samples to achieve satisfactory segmentation performance. And this severely hinders the performance improvement of most of existing augmentation methods. To tackle this problem, in this paper, we propose an object-level remote sensing image augmentation approach based on leveraging the U-Net-based generative adversarial networks. Specifically, our proposed approach consists two components including the semantic tag image generator and the U-Net GAN-based translator. To evaluate the effectiveness of the proposed approach, comprehensive experiments are conducted on a public dataset HRSC2016. State-of-the-art generative models, DCGAN, WGAN, and CycleGAN, are selected as baselines. According to the experimental results, our proposed approach significantly outperforms the baselines in terms of not only drawing the outlines of target objects but also capturing their meaningful details.

1. Introduction

With the continuous development of satellite remote sensing image technology, high-resolution satellite image makes the target segmentation technology of satellite image realized. In many fields, the segmentation of satellite image can help to collect information and collect information quickly. Different from other image segmentation, satellite image contains a large number of elements and is easily affected by weather and season, so it needs a large number of datasets for training; otherwise, the target model will have difficulty in learning the relevant feature distribution. Especially, for the ship remote sensing image, because the ship is usually in dynamic change, it is difficult to collect a lot of data from the target, so it is necessary to augment the dataset. For semantic segmentation task, when training a model, paired data is needed, that is, an original image and an image with semantic tag. Therefore, we need to construct two corresponding images at the same time.

For traditional data augmentation methods such as CutOut [1], the input square region is randomly masked in the training process, which can improve the robustness and overall performance of the convolutional neural network. CutMix [2] generates a new training sample by randomly combining two trimmed training samples, which makes its performance better than CutOut. However, these methods generate new samples directly modified and then stitched at the image level, which means that the boundaries among different objects cannot be clearly identified. Since boundaries are of vital importance in the semantic segmentation task, the above-mentioned methods are not suitable for augmenting samples targeting at semantic segmentation task.

In recent years, the concept of generative adversarial networks [3] (GAN) has become one of the most popular unsupervised algorithms. For instance, DCGAN [4] and Marta GAN [5] have been proposed to augment remote sensing images. However, due to the complexity and uncertainty presented in remote sensing images, it is difficult for GAN-based augmentation methods to learn the distribution characteristics of the target objects, resulting in unsatisfactory augmentation effect. For example, the resolution of the generated images is limited while most of meaningful details are missed. Moreover, GAN-based augmentation methods cannot be able to generate the paired semantic tag images which are critical to enable semantic segmentation task and usually annotated manually with high cost. Therefore, it is desired to propose an approach to augment remote sensing images by effectively tackling the complexity and reducing the annotation cost.

Recently, conditional GANs [6] are proposed, which is a variant of GANs and capable of performing the image translation task. Inspired by conditional GANs, we propose an approach consisting two main components including the semantic tag image generator and the translator to augment remote sensing images. Firstly, the target objects are extracted by learning original training samples and then reasonably composed to construct the semantic tag image. Secondly, the translator based on U-Net [7] GANs is responsible of transforming the generated semantic tag images into realistic-looking images (please refer to Section 4 for more details).

In this work, our contributions could be summarized as follows:

(i) A framework based on U-Net GANs for remote sensing image augmentation is proposed in this paper.

(ii) A new method to automatically generate semantic tag images is proposed with a set of heuristic generation rules and restriction rules.

(iii) Comprehensive experiments are conducted on a public remote sensing image dataset while in-depth analysis is provided focusing on the comparison between our proposed approach and baselines.

The rest part of this paper is organized as follows. Concrete examples about remote sensing image augmentation and basics of both GAN and U-Net models are offered in Section 2. The remote sensing image augmentation problem is formally defined in Section 3. In Section 4, the methodology is illustrated in detail including the overall architecture of the proposed approach, semantic tag image generator, and remote sensing image translator. In Section 5, the experiments are conducted on a public remote sensing image dataset to validate the effectiveness of the proposed approach. Related works about existing remote sensing image augmentation solutions are discussed in Section 6 followed by the conclusion provided in Section 7 to summarize this work.

2. Preliminaries

2.1. Examples of Remote Sensing Image Augmentation. Different from the data augmentation task directly performed at the image scale, for semantic segmentation tasks, they mainly focus on differentiating target objects from the background. The reason that traditional image-level augmentation cannot well support semantic segmentation tasks is due to its inability to identify the features of target objects, or the boundary between target objects and the background. Therefore, it becomes the motivation for the work in this paper to propose an approach to augment the original images at the object level rather than the image level.

To further illustrate the difference between image-level and object-level remote sensing image augmentation, examples are provided as shown in Figure 1. The upper row of Figure 1 shows the typical augmentation operations such as crop, flipping, cutout, and stretch usually adopted in image-level augmentation. And the lower row of Figure 1 presents object-level augmentation operations including object remove, object flipping, cutout without destroying the integrity of the original object, and semantically reasonable object add. Obviously, object-level remote sensing image augmentation can better serve the semantic segmentation task by flexibly composing different objects into the newly generated images, compared with the image-level counterpart.

2.2. Basics of Generative Adversarial Networks. Generative adversarial networks (GANs) were introduced in 2014 [3] and widely applied to various application scenarios [5, 8, 9]. GAN is able to produce high-quality output images through the mutual game learning of (at least) two independent modules: the generative model and the discriminative model.

(1) Generative model (aka Generator) has the goal of capturing the data distribution from training samples by receiving a random noise z and generating an image from that noise, which is denoted as $G(z)$

(2) Discriminative model (aka Discriminator) has the task of telling if the current sample comes from the training set or from the Generator. Its input parameter is x, which may be extracted from the training sample or the "fake" sample generated by the Generator. Its output is 1 or 0, while 1 indicates that the Discriminator judges the sample as the real sample and 0 means that the Discriminator judges the sample as the fake sample.

The generative model G aims to learn a distribution p_g over data x, by building a mapping function from a prior noise distribution $p_z(z)$ to the data space, $G(z ; \theta_g)$, where θ_g are the parameters of the model G, e.g. the weights of the multilayer perceptrons to implement G.

The discriminative model $D(x ; \theta_d)$ is an independent module to be implemented as a binary classifier, which outputs a single scalar (i.e., 0 or 1) representing the probability that x came form the training set rather than p_g.

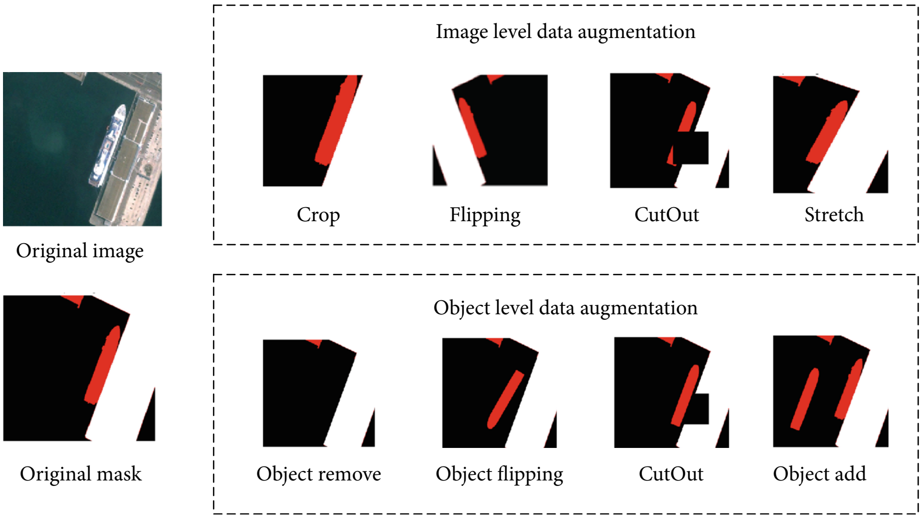


Figure 1: Examples of image-level and object-level remote sensing image augmentation.

Then, both models are jointly trained to play the following two-player min-max game as defined in the following equation until they reach the Nash Equilibrium:

$$\min_{G}\max_{D} V(D,G)=E_{x\sim p_{\text{data}}(x)}[\log D(x)] + E_{z\sim p_z(z)}[\log(1-D(G(z)))]. \quad (1)$$

2.3. Basics of U-Net. In the last decade, deep learning models have been universally applied to different application scenarios such as time series analytics [10], rate adaptation [11], and edge computing [12]. As a dedicated deep learning model, U-Net has made great successes on the semantic segmentation task of medical images [7]. Therefore, it is chosen as the basic component of the model employed in this work for object-level remote sensing image augmentation. U-Net is named due to its symmetrical structure looks like an upper letter U as shown in Figure 2. For an input image with the size $N \times N$, U-Net will firstly conduct the 3×3 convolution for twice as shown in the upper left component of Figure 2. And then, the max pooling will be executed to downsize the output of the last layer of the upper component to fit the size of the first layer of the lower component, which is shown as the red downward arrow. The above procedure will be iterated for 4 times until the bottom component is reached, which plays the role as the conjunction of the left and right edges of U-Net. On the right edge of U-Net, the 3×3 convolution will be done within each component. Different from the operations among components of the left edge, the 2×2 upconvolution is conducted to restore the downsized sample to its original size. Moreover, the gray horizontal arrow represents the copy and crop operations on the output of the last layer of the left edge component that is taken as part of the input of the first layer of the corresponding right edge component. This feature is regarded as one of the factors making U-Net so successful on semantic segmentation of medical images, which embeds multiscale grain of the input image into the learning process.

3. Problem Formulation

The object-level remote sensing image augmentation problem could be formulated as follows.

The following are given:

(i) $I=\{p\}^{W\times H}$, where I is a remote sensing image, p is a pixel, and W and H mean the width and height of the image

(ii) $T=\{t_n\}^{W\times H}$, where T is a semantic tag image and t is a pixel with the tag n

(iii) $S=\{I_k, T_k\}$, where S is the training set with K paired original image I_k and its semantic tag image T_k.

Assume that

(i) The pixels p and t at the same position of the paired I_k and T_k indicate the same object

(ii) Each object identified in T_k consists a set of pixels which are spatially connected

(iii) There exists at least one mapping function from T_k to I_k, which not only draws the outlines of objects of T_k but also captures their meaningful details to be presented in I_k.

The objective is as follows:

(i) By learning the mapping function from T_k to I_k, it aims to generate a set of synthetic remote sensing images $A=\{I_m, T_m\}$ which are of higher diversity and reasonably realistic looking.

In the following section, we would like to introduce an approach to augment the remote sensing images at the object level by leveraging the generative adversarial network architecture based on U-Net.

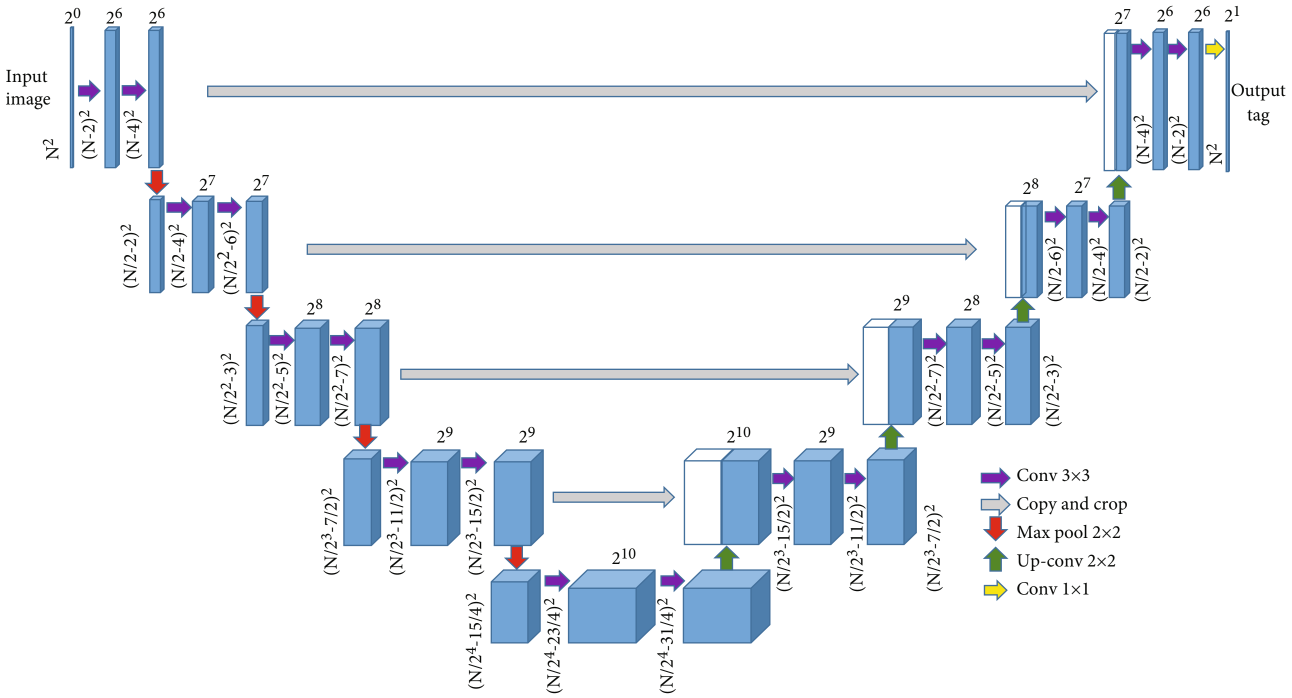


Figure 2: Structure of U-Net.

4. Methodology

4.1. Overall Architecture. As shown in Figure 3, the proposed approach to augmenting remote sensing image at the object level is composed of two key components. The first key component is the semantic tag image generator, while the other is the translator based on U-Net GANs. At the very beginning of the whole process, the semantic tag image generator takes the original training set as the input to identify different types of the objects in a pixel-wise manner. And then, those objects could be flexibly composed into a tag image subject to the predefined constraints. After that, by taking original training set and tag images as the input, the translator based on U-Net GANs is responsible of generating remote sensing images. Finally, the new training set for semantic segmentation task is obtained by integrating the original training set with both the generated images and their corresponding tag images. The design details about the semantic tag image generator and translator based on U-Net GANs are illustrated in detain in Section 4.2 and Section 4.3.

4.2. Semantic Tag Image Generator. The semantic tag image generator is aimed at identifying and extracting the target objects as pure color regions from original images of the training set. Each color represents one specific type of the target objects. Since we are more interested in remote sensing images mostly containing the port areas, three types of the target objects will be automatically identified and extracted including the water surface, the port land, and ships as shown in Figure 4. Besides the identification and extraction of target objects, out proposed semantic tag image generator is able to automatically compose the identified objects into tag images in a harmonic manner under the guidance of generation rules and restriction rules which are presented in detail as follows.

The overall procedure for generating semantic tag images is shown in Figure 4. The generation rules are listed as follows.

(i) The black-colored region is generated as the background which is usually the water surface around the port land or the ships

(ii) The white-colored region is composed of a set of randomly generated white pixels which are adjacent to each other

(iii) Ship tags are learned from the training set and placed to the proper black color regions subject to the restriction rules that are stated as below.

In order to ensure the reasonability of the generated images, it is required to make sure that the special layout among different objects of the tag image is proper and reasonable. Hence, three restriction rules on the placement of all objects are proposed as follows.

(i) There is no overlapping between any two ships, i.e., no overlapping between any two red-colored regions in the generated tag image

(ii) There is no overlapping between any ship and the port land, i.e., no overlapping between any red color region and the white-colored region in the generated tag image

(iii) There is at least one ship but no more than the maximum number of ships observed in the training set.

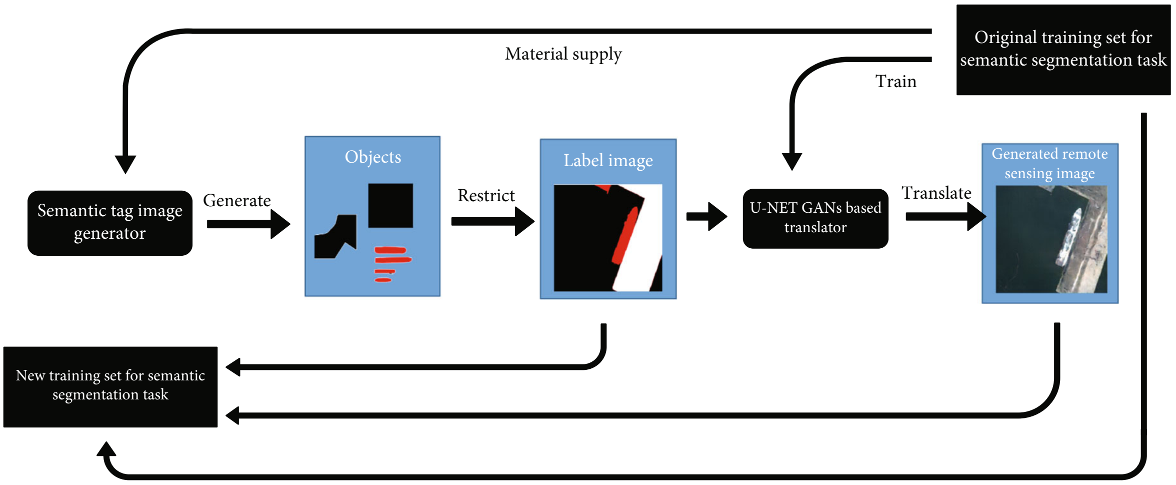

Original training set for semantic segmentation task
Material supply
Train
Semantic tag image generator
Generate
Objects
Restrict
Label image
U-NET GANs based translator
Translate
Generated remote sensing image
New training set for semantic segmentation task

Figure 3: Overall architecture of the proposed approach to object-level remote sensing image augmentation.

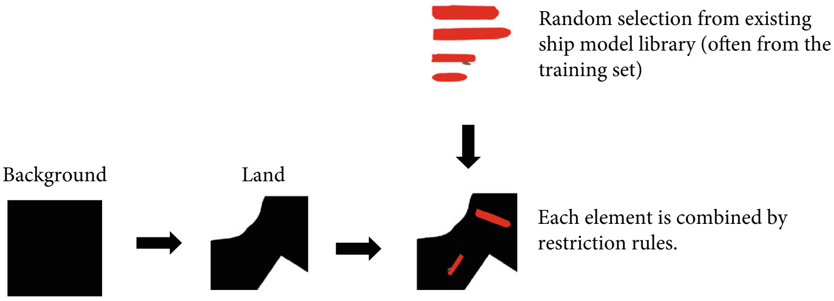

Random selection from existing ship model library (often from the training set)
Background
Land
Each element is combined by restriction rules.

Figure 4: Procedure for generating semantic tag image.

By applying the above introduced generation rules and restriction rules to the generation procedure of tag images, proper tag images with typical objects flexibly placed could be trivially obtained. Afterwards, given the generated tag images, we propose a translator based on U-Net GANs to transform each tag image to a synthetic remote sensing image by learning pixel-wise details from the original training set. And the second use of the tag image is to take them as the ground truth for training and testing the semantic segmentation model. The design details of the translator are provided in Section 4.3.

4.3. U-Net GAN-Based Translator. Given the generated tag image, the translator is responsible in transforming it to a synthetic but realistic-looking remote sensing image. In this paper, we employ the generative adversarial networks as the reference for implementing the translator. As shown in Figure 5, the translator consists two key components including a generative model Generator and a discriminative model Discriminator. The design of the Generator conforms to that of U-Net introduced in Section 2.3. And the design of Discriminator is a FCN model as proposed by [13]. The training procedure for U-Net GAN-based translator follows the steps stated as below.

(i) Step 1: Generator takes the semantic tag image as the input which is denoted as z

(ii) Step 2: Generator generates a new remote sensing image denoted as IMG' which tries to deceive the Discriminator for judging IMG' as the real remote sensing image

(iii) Step 3: Discriminator makes every endeavour to figure out whether IMG' is real or fake according to original images IMG from the training set

(iv) Step 4: the above three steps will be executed in an iterative manner until the Nash Equilibrium is reached.

The optimization goal of U-Net GAN-based translator consists two parts. The Generator denoted as G for short needs to learn a distribution p_g over output images x by building a mapping function $G(z;\theta_g)$ from the given tag image distribution $p_z(z)$ to the original image representation space. θ_g are the parameters of the Generator, i.e., the weights of U-Net implementing Generator in this paper. And the Discriminator denoted as D for short is implemented as a binary classifier, which outputs a single scalar

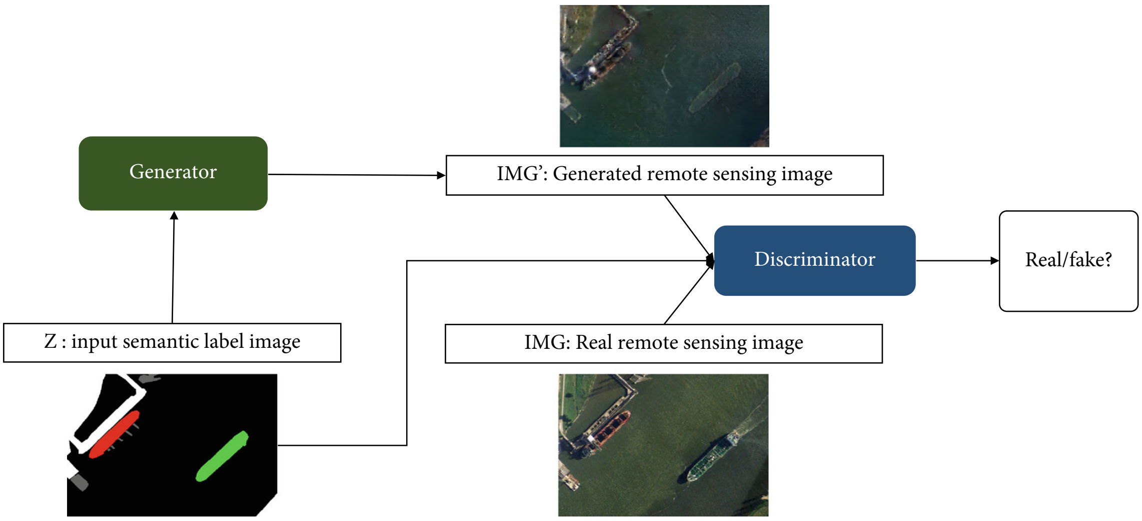

Generator
IMG’: Generated remote sensing image
Discriminator
Real/fake?
Z : input semantic label image
IMG: Real remote sensing image

Figure 5: Model architecture of U-Net GAN-based translator.

representing the probability that output images x of the Generator came from the training set rather than p_g.

The loss function $L(G)$ of the Generator$G$ could be mathematically defined as shown in the following equation.

$$L(G) = E_{z\sim p_z(z)}[\log(1 - D(z, G(z)))]. \tag{2}$$

The loss function $L(D)$ of the *DiscriminatorD* could be mathematically defined as shown in Equation (3).

$$L(D) = E_{x\sim p_{data}(x)}[\log D(z, x)] \tag{3}$$

Generally, the optimization objective of the translator could be defined as shown in the following equation:

$$\min_G \max_D V(D, G) = L(D) + L(G). \tag{4}$$

In order to let the generator better learn the details of target images, it is beneficial to integrate the traditional GANs’ optimization objective with an extra loss such as the smooth $L1$ distance. The smooth $L1$ distance $L_{smooth\,L1}(G)$ is mathematically defined as shown in the following equation:

$$L_{\text{smooth}\,L1}(G) = \begin{cases} E_{z,x}\left[0.5 * (x - G(z))^2\right] & \|(x - G(z))\| < 1 \\ E_{z,x}[\|(x - G(z))\| - 0.5] & \text{otherwise.} \end{cases} \tag{5}$$

Meanwhile, the role of the Generator$G$ has changed to not only fool the Discriminator$D$ but also approach multiscale grains of the ground-truth images guided by the new optimization objective as defined by the following equation:

$$\min_G \max_D V(D, G) = L(D) + L(G) + L_{\text{smooth}\,L1}(G) \tag{6}$$

5. Evaluation

5.1. Experimental Settings. A public dataset called HRSC2016 [14] is adopted to evaluate the effectiveness of the proposed U-Net GAN-based approach to remote sensing image augmentation. In the dataset HRSC2016, all the images are collected from six famous harbors with the resolutions ranging from 0.4 m to 2 m. The image sizes vary from 300 to 1500 while most of them are larger than 1000 × 600. The training set contains 436 images including 1207 samples, and the validation set contains 181 images including 541 samples, respectively. The test set contains 444 images including 1228 samples.

As for the baselines, we are going to compare our approach with two typical types of augmentation methods including the geometric transformation methods and generative models. Specifically, four types of transformations including Scaling, Flipping, CutOut, and CutMix will be tested for evaluation. Moreover, three generative models including WGAN [15], DCGAN, and CycleGAN [16] will also be evaluated under the same evaluation settings. The key hyperparameter settings of the baseline models and our proposed model are shown in Table 1.

All the experiments are conducted on a Windows 10 64-bit server equipped with one Intel Xeon CPU at 3.7 GHz and 64 GB main memory at 2666 MHz. All the generative models are trained on one NVIDIA GeForce RTX 2080Ti GPU of which the dedicated memory is 11 GB. And the deep learning framework to support the implementation and training of generative models is tensorflow 2.3 library in the Python 3.8 environment.

5.2. Experimental Results and Analysis. Firstly, we would like to compare the performance of geometric transformation methods with the approach proposed in this work. As shown in Figure 6, given a pair of the original image and its tag

Table 1: Key hyperparameter settings of baselines and our model.

Model	Key hyperparameter settings

DCGAN	Epochs = 300, batch size = 16, activation Func. = tan h
	Optimizer = Adam, loss Func. = binary cross entropy

WGAN	Epochs = 300, batch size = 16, activation Func. = tan h
	Optimizer = Adam, loss Func. = binary cross entropy

CycleGAN	Epochs = 300, batch size = 16, activation Func. = tan h Optimizer = Adam, loss Func. = binary cross entropy, lambda = 10

Ours	Epochs = 300, batch size = 16, activation Func. = tan h
	Optimizer = Adam, loss Func. = binary cross entropy, LAMBDA = 100

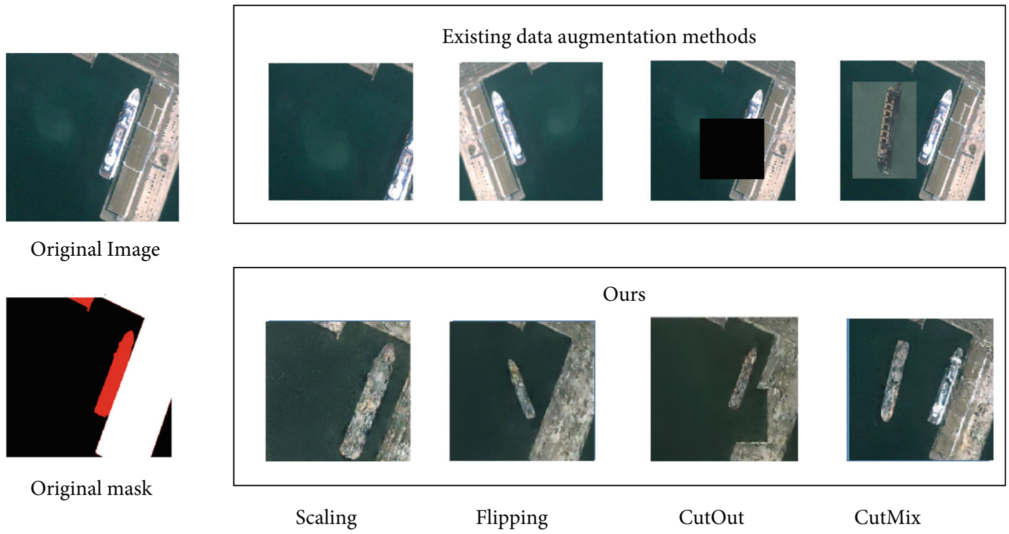


Figure 6: Comparison with existing geometric transformation methods.

image, the upper row lists the augmented images after Scaling, Flipping, CutOut, and CutMix by leveraging traditional geometric transformation methods while the lower row shows the augmented images generated by our proposed approach. In the "Scaling" case, the target object (i.e., the ship) is partially cut while it is simultaneously scaled with the whole image by our approach. In the "Flipping" case, our approach turns the direction of the ship instead of simply doing the vertical flipping as done by traditional geometric transformation methods. In the "CutOut" case, the augmented image has a very inharmonious black region which is smartly processed by our proposed approach. More interesting, our approach patches the black region with the water surface while maintains the integrity of the ship. At last, in the "CutMix" case, the traditional geometric transformation methods simply place a rectangle patch containing another ship with considering the semantic consistency between the patch and the original image. In contrast, our proposed approach places the newly added ship to the proper area of the original image with its background seamlessly wired. According to the above analyzed cases, it is clear that our proposed approach significantly outperform the traditional geometric transformation methods in terms of maintaining object integrity, diversity, and background harmony for the augmented remote sensing images.

In another set of experiments, we compare the performance of different generative models including DCGAN, WCGAN, CycleGAN, and our proposed approach. As shown in the first two rows of Table 2, DCGAN and WGAN only accept random noise as the input and can hardly generate meaningful output images. The most competitive generative model is CycleGAN of which the generation results are shown in the third row of Table 2. Obviously, CycleGAN is able to generate the outline for each type of target objects (i.e., the water surface, port land, and ships). However, if we zoom in the images generated by CycleGAN, it is found that almost no detail of the target objects is captured. And our proposed approach does not only draw the outline of multiple objects but also capture their much more details in the generated images.

Furthermore, the detailed training process of each generative model is shown in Table 3. It is observed that DCGAN and WGAN are not able to generate images rather than random noise until Epoch 200. And even after Epoch 200, DCGAN and WGAN just capture some very vague features which cannot be clearly identified. CycleGAN and our

Table 2: Comparison with existing generative models.

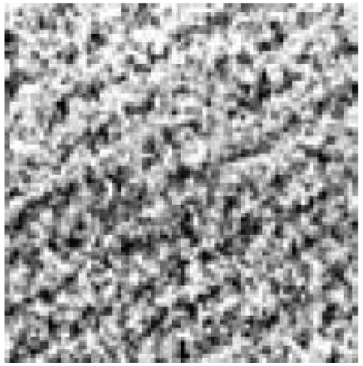
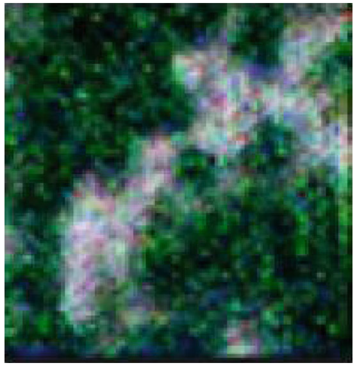
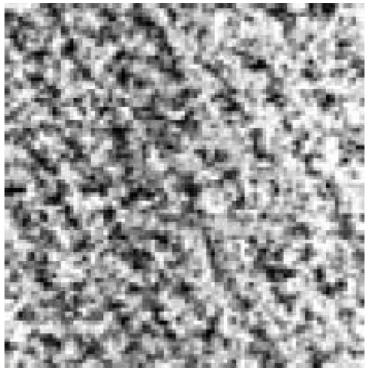
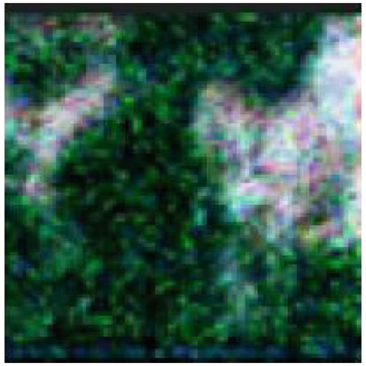
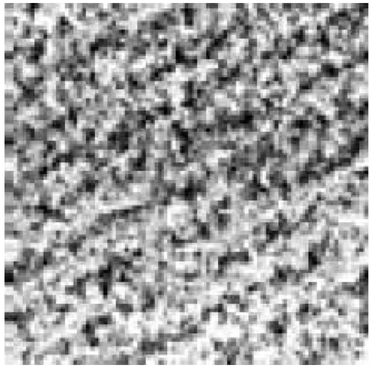
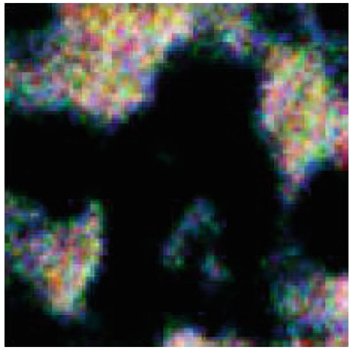
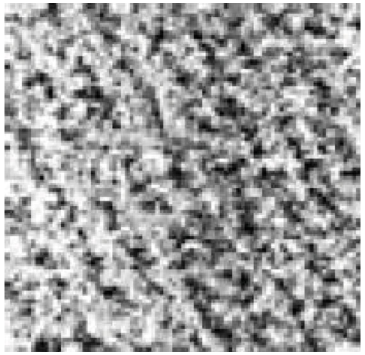
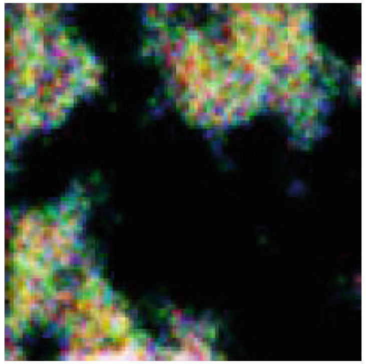
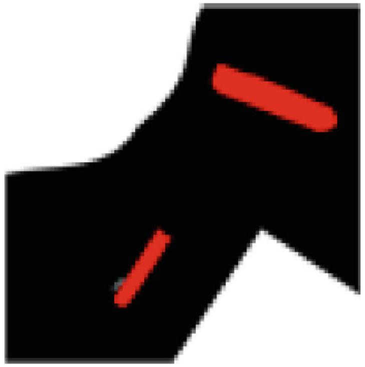
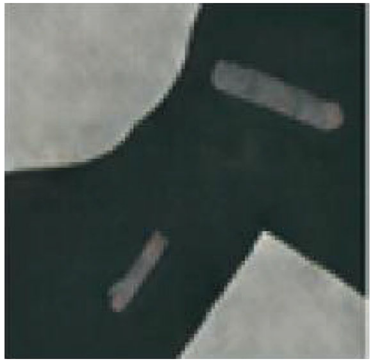
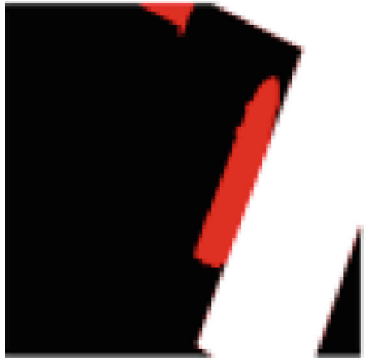
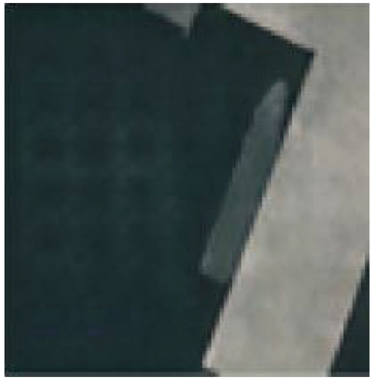
Model	Input1	Output1	Input2	Output2
DCGAN				
WGAN				
CycleGAN				

Table 2: Continued.

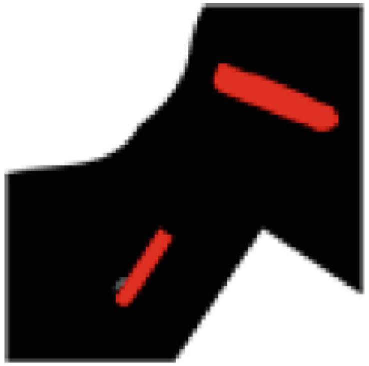
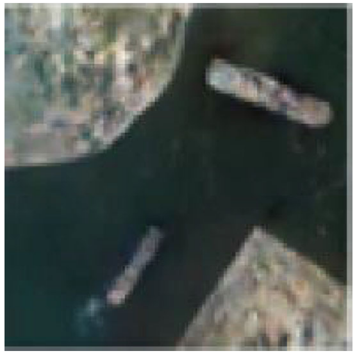
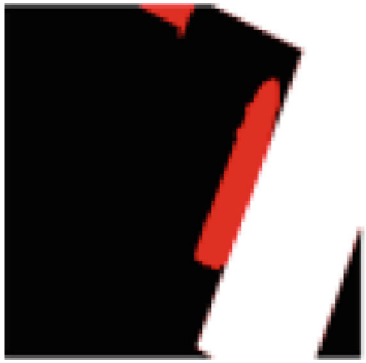
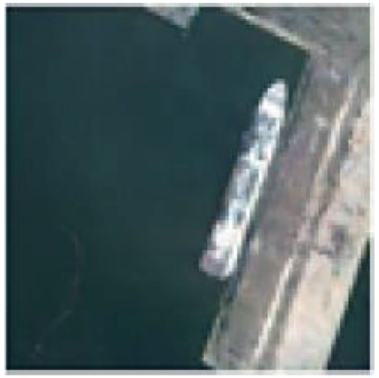
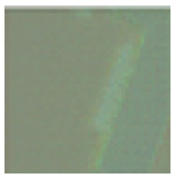
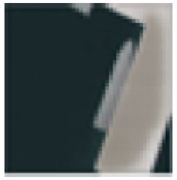
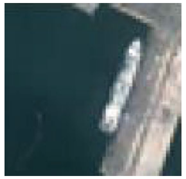
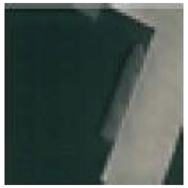
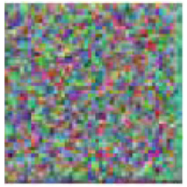
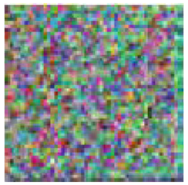
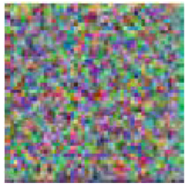
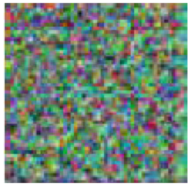
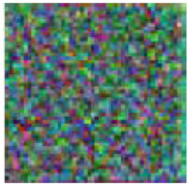
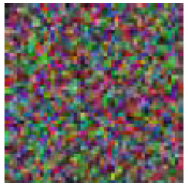
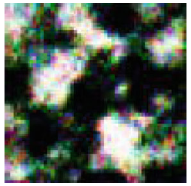
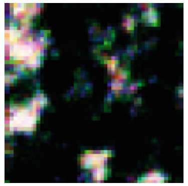
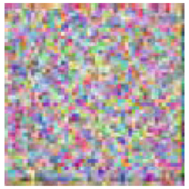
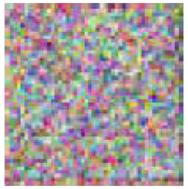
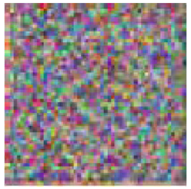
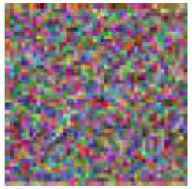
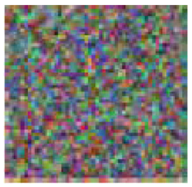
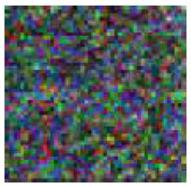
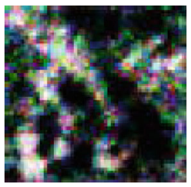
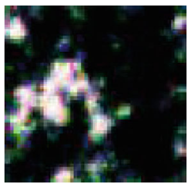
Model	Input1	Output1	Input2	Output2
Ours				

Table 3: Detailed training process of baselines and our model.

Model	0	5	10	15	20	100	200	300
Ours								
CycleGAN								
DCGAN								
WGAN								

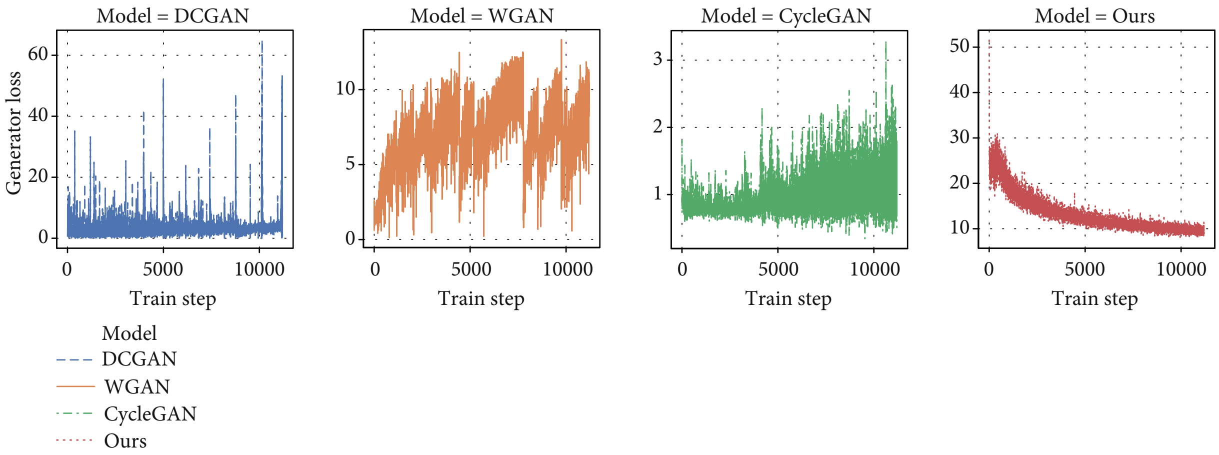


Figure 7: Generator loss of baselines and our model over train steps.

proposed approach start the training with the similar pure color input. Both models are able to capture the outline of each object as early as Epoch 5. But as the training epoch elapsed, CycleGAN is unable to capture more details about outlined objects while our proposed approach gradually adds more details to those objects. And finally, the output image generated by our proposed approach at Epoch 300 shows the highest visible similarity with the ground-truth image among all the generative models. According to the experimental results listed above, in summary, it is validated that our proposed approach to remote sensing image augmentation significantly outperforms the baselines including the traditional geometric transformation methods and generative models.

Last but not the least, the generator loss of all baselines and our model over each train step is shown in Figure 7 so that we can observe the learning behavior of the generator of all baseline models and our proposed model. And it can be clearly observed that the loss value of DCGAN's generator fluctuates from the very beginning of the training process till the last train step. For WGAN, the loss value of its generator becomes higher and higher over train steps. It indicates the fact that the generator of both DCGAN and WGAN can hardly converge on the experimental dataset, and thus, no meaningful image could be generated. As for CycleGAN, it performs better than DCGAN and WGAN by showing a converging trend during the training process. However, its loss value has a high deviation which probably means its generator cannot learn complex features from original images in a stable manner. At last, when we analyze the loss value of the generator of our proposed model, it presents a much better converging trend over train steps than baseline models. And this is a strong evidence to confirm the superiority of our proposed model over baselines in the task of object-level remote sensing image augmentation.

6. Related Work

Data augmentation is the technique to augment original training samples by generating new samples. The existing data augmentation techniques can be roughly divided into the following two categories: (1) geometric transformation methods, which generate new samples by performing various geometric operations on original samples, and (2) generative models, which generate new samples by learning discriminative features of original samples and utilizing their labels.

Geometric transformation methods have been widely used, including random cropping, horizontal flipping, and color enhancement [17], which can improve the robustness of translation and reflection and illumination objects, respectively. Random scaling, random rotation, and affine transformation are also widely used in data augmentation scenarios [1]. Moreover, CutOut and CutMix [2] are also employed to augment new samples by learning features from original samples. In general, geometric transformation methods are usually applied to solve either the class imbalance problem or the limited sample problem. According to previous studies [1, 2, 17], the above-mentioned methods have been proved to be fast, reproducible, and reliable. And their implementation is relatively simple, which can be easily generalized to the currently popular deep learning framework. However, these methods can only perform image-level transformation, which means they only change the depth or scale of the image after generation. Although image-oriented tasks such as image classification benefit from geometric transformation methods, they are not capable of improving object-oriented tasks such as the semantic image segmentation.

Despite the many successes of generative adversarial network (GAN) and its numerous variants, there are still a lot of challenging issues such as mode collapse [8] and generation quality [18, 19]. Objectaug [20] is one kind of generative models for object-level data augmentation. It decomposes the image into separate objects and backgrounds using semantic tags and applies augmentation on background and objects individually. Objectaug can effectively enhance the boundaries between the target objects and the background. However, its core data augmentation method is still based on the traditional geometric transformation, which limits the diversity of generated samples. Conditional adversarial nets [6] are proposed to handle both unimodal and multimodal samples by extending the original GAN to its conditional variant. As for other GAN-based models like DCGAN [4] and WGAN [15], they are not capable of generating new samples with visually similar features as those of the original samples due to the lack of properly guided input. Different from DCGAN and WGAN, CycleGAN [16] incorporates additional information with the original input which greatly enhances the quality of the generated samples. However, the main drawback of CycleGAN is its unpaired training process which limits its further performance improvement.

By taking drawbacks of the aforementioned data augmentation methods into accounts, in this paper, it motivates us to design and implement a new approach to augmenting existing dataset by generating diverse and high-quality samples at the object level.

7. Conclusion

In this paper, we study the object-level remote sensing image augmentation problem. In Section 3, the problem formulation is provided in a formal format to facilitate the understanding of the target problem. Then, an approach composed of the semantic tag image generator and the U-Net GAN-based translator is proposed in Section 4 to illustrate in detail how we can achieve object-level remote sensing image augmentation. To validate the effectiveness of the proposed approach, comprehensive experiments are conducted on a public dataset HRSC2016. With experimental results carefully examined and analyzed in Section 5.2, our proposed approach shows the promising performance by not only drawing the outline of different objects but also capturing their meaningful details.

Data Availability

The dataset used to support the evaluation of the proposed approach is available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research is supported by the Infrastructure Construction Project of Naval University of Engineering (No. HCX-MN-0600006).

References

[1] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2015, https://arxiv.org/abs/1409.1556.

[2] S. Yun, D. Han, S. Chun, S. J. Oh, Y. Yoo, and J. Choe, "CutMix: regularization strategy to train strong classifiers with localizable features," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 6022–6031, Seoul, Korea (South), 2019.

[3] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza et al., "Generative adversarial networks," *Advances in Neural Information Processing Systems*, vol. 3, pp. 2672–2680, 2014.

[4] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," 2015, https://arxiv.org/abs/1511.06434.

[5] D. Lin, K. Fu, Y. Wang, G. Xu, and X. Sun, "MARTA GANs: unsupervised representation learning for remote sensing image classification," *IEEE Geoscience and Remote Sensing Letters*, vol. 14, no. 11, pp. 2092–2096, 2017.

[6] M. Mirza and S. Osindero, "Conditional generative adversarial nets," 2014, https://arxiv.org/abs/1411.1784.

[7] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention (MICCAI 2015)*, N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, Eds., pp. 234–241, Springer International Publishing, Cham, 2015.

[8] W. Li, L. Fan, Z. Wang, C. Ma, and X. Cui, "Tackling mode collapse in multi-generator GANs with orthogonal vectors," *Pattern Recognition*, vol. 110, p. 107646, 2021.

[9] W. Li, L. Xu, Z. Liang et al., "Sketch-then-edit generative adversarial network," *Knowledge-Based Systems*, vol. 203, p. 106102, 2020.

[10] C. Ma, X. Shi, W. Zhu, W. Li, X. Cui, and H. Gui, "An approach to time series classification using binary distribution tree," in *2019 15th International Conference on Mobile Ad-Hoc and Sensor Networks (MSN)*, pp. 399–404, Shenzhen, China, 2019.

[11] W. Xu, C. Ma, S. Guo, and H. Zhou, "Efficient rate adaptation for 802.11af TVWS vehicular access via deep learning," in *2019 IEEE Global Communications Conference (GLOBECOM)*, pp. 1–6, Waikoloa, HI, USA, 2019.

[12] C. Ma, X. Shi, W. Li, and W. Zhu, "Edge4tsc: binary distribution tree-enabled time series classification in edge environment," *Sensors*, vol. 20, no. 7, p. 1908, 2020.

[13] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3431–3440, Boston, MA, USA, 2015.

[14] Z. Liu, L. Yuan, L. Weng, and Y. Yang, "A high resolution optical satellite image dataset for ship recognition and some new baselines," in *6th International Conference on Pattern Recognition Applications and Methods (ICPRAM 2017)*, pp. 324–331, Porto, Portugal, 2017.

[15] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *34th International Conference on Machine Learning (ICML)*, pp. 214–223, Sydney, Australia, 2017.

[16] J. Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *2017 IEEE International Conference on Computer Vision (ICCV)*, Venice, Italy, 2017.

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[18] W. Li, W. Ding, R. Sadasivam, X. Cui, and P. Chen, "His-GAN: a histogram-based GAN model to improve data generation quality," *Neural Networks*, vol. 119, pp. 31–45, 2019.

[19] W. Li, Z. Liang, P. Ma, R. Wang, X. Cui, and P. Chen, "Hausdorff GAN: Improving GAN generation quality with Hausdorff metric," *IEEE Transactions on Cybernetics*, pp. 1–13, 2021.

[20] J. Zhang, Y. Zhang, and X. Xu, "Objectaug: object-level data augmentation for semantic image segmentation," 2021, https://arxiv.org/abs/2102.00221.

Hindawi
Wireless Communications and Mobile Computing
Volume 2021, Article ID 7181815, 7 pages
https://doi.org/10.1155/2021/7181815

Research Article

Mol-BERT: An Effective Molecular Representation with BERT for Molecular Property Prediction

Juncai Li[1] **and Xiaofei Jiang**[2]

[1]*Hunan Vocational College of Electronic and Technology, Changsha 410220, China*
[2]*College of Information Science and Engineering, Hunan University, Changsha 410082, China*

Correspondence should be addressed to Xiaofei Jiang; jiangxiaofei@hnu.edu.cn

Received 30 June 2021; Accepted 13 August 2021; Published 3 September 2021

Academic Editor: Yulin Wang

Molecular property prediction is an essential task in drug discovery. Most computational approaches with deep learning techniques either focus on designing novel molecular representation or combining with some advanced models together. However, researchers pay fewer attention to the potential benefits in massive unlabeled molecular data (e.g., ZINC). This task becomes increasingly challenging owing to the limitation of the scale of labeled data. Motivated by the recent advancements of pretrained models in natural language processing, the drug molecule can be naturally viewed as language to some extent. In this paper, we investigate how to develop the pretrained model BERT to extract useful molecular substructure information for molecular property prediction. We present a novel end-to-end deep learning framework, named Mol-BERT, that combines an effective molecular representation with pretrained BERT model tailored for molecular property prediction. Specifically, a large-scale prediction BERT model is pretrained to generate the embedding of molecular substructures, by using four million unlabeled drug SMILES (i.e., ZINC 15 and ChEMBL 27). Then, the pretrained BERT model can be fine-tuned on various molecular property prediction tasks. To examine the performance of our proposed Mol-BERT, we conduct several experiments on 4 widely used molecular datasets. In comparison to the traditional and state-of-the-art baselines, the results illustrate that our proposed Mol-BERT can outperform the current sequence-based methods and achieve at least 2% improvement on ROC-AUC score on Tox21, SIDER, and ClinTox dataset.

1. Introduction

Effectively identifying the molecular properties (e.g., bioactivity and toxicity) plays an essential part in drug discovery and material science, which can alleviate the costly and time-consuming process in comparison to the traditional experiment methods [1]. Such a process is usually known as molecular property prediction, and it is a fundamental task to explore the functionality of new drugs. A typical molecular property prediction system takes the drug features of descriptors as the input and outputs the predicted result of predefined chemical properties. The predicted value can benefit various subsequent tasks, including virtual screening [2–4] and drug repurposing [5–7]. However, accurately predicting molecular property with computational methods remains challenging.

Previous machine learning approaches focused on designing a variety of expert-engineered descriptors or molecular fingerprints manually based on experimental statistics to predict molecular property [8–10]. For example, extended-connectivity fingerprint (ECFP) [11], as the most representative fingerprint method, was designed to generate different types of circular fingerprints that extracted the molecular structures of atom neighborhoods by using a fixed hash function [12]. Then, these obtained fingerprint representations would be sent to traditional machine learning models to perform further predictions, and it can be applied to a wide range of different models, such as logistic regression, support vector classification, kernel ridge regression, random forest, influence relevance voting, and multitask networks [13]. However, this line of researches heavily depends on the design of hand-crafted features and domain knowledge. Besides, the generated hash bit vectors are difficult to biologically understand the relationship between chemical properties and molecular structures.

Inspired by the remarkable achievements that deep learning has shown in a variety of domains, including computer vision [14] and natural language processing [15, 16], it also has gained lots of attention for molecular property prediction. The molecular representation methods being introduced can be mainly summarized into two parts: sequence-based and graph-based approaches. For sequence-based methods, simplified molecular input line entry specification, shortened as SMILES, is the most common molecular linear notation that encodes the molecular topology on the basis of chemical rules [17]. In this way, several methods are attempted to take SMILES representation as the input and use current successful models (e.g., recurrent neural networks) to obtain molecular representations [18], while this line of work suffered from insufficient labeled data for specific molecular tasks. More recently, researchers adopted the unsupervised and pretraining strategies in natural language processing (NLP) to learn contextual information from large unlabeled molecular datasets. For example, an unsupervised machine learning method named Mol2vec was developed to learn vector representations of molecular substructures [19]. And SMILES-BERT was proposed to pretrain the model through a masked SMILES recovery task by designing attention mechanism-based transformer layer [20]. These pretrained methods pay more attention to the contextual information of molecular sequences, but they hardly consider some molecular substructure (i.e., functional groups) that essentially contributes to the molecular property [21, 22].

On the other hand, graph neural networks (GNNs) have been adopted to explore the graph-based representation for molecular property prediction [23–25]. Graph convolutions were the first work that applied the convolutional layers to encode molecular graph into neural fingerprints [26]. Similarly, much efforts are made to extend a variety of GNNs on property prediction tasks. For example, the weave featurization encoded chemical features to form molecule-level representations [27]. And some methods extended graph attention network [28] to learn the aggregation weights [25, 29]. Moreover, to better encode the interactions between atoms, a message passing neural network named MPNN was designed to utilize the attributed features of both atoms and edges [30]. More recently, DMPNN [31] and CMPNN [32] were further introduced to leverage the attributed information of nodes and edges during message passing. Although graph-based models have achieved great performance on molecular graph representation, they seldom make use of the vast available biological sequence data.

Recently, substantial pretrained models [33–37] trained on the large corpus or unlabeled data can learn universal representations, which are benefit for various downstream tasks, including protein sequence representation [38, 39], biomedical text mining [40, 41], and chemical reaction prediction [42]. Advances in pretrained models have shown their powerful ability for extracting information from unlabeled sequences, which raises a tantalizing question: can we develop a pretrained model to extract useful molecular substructure information from massive SMILES sequence datasets? To help solve this problem, we propose a novel neural framework, named Mol-BERT, tailored for molecular property prediction. The idea of Mol-BERT is natural and intuitive. Our framework consists of three types of modules. The feature extractor is first to extract atom-level and substructure features centered on the current atom, and the first module can be replaced with a wide range of different molecular representation methods. Then, the pretrained BERT module learns molecular substructure or fragment information from large pretraining corpus (i.e., unlabeled SMILES sequences). The final module is to predict the specific molecular property after fine-tuning the pretrained Mol-BERT via a multityped classifier. To illustrate the performance of our proposed method in various prediction tasks, Mol-BERT is fine-tuned and evaluated on 4 widely used molecular benchmark datasets. In comparison to state-of-the-art baselines (i.e., sequence- and graph-based methods), the experimental results prove the effectiveness of our proposed Mol-BERT.

This paper is organized as follows. Section 2 firstly introduces the preprocessed corpus for Mol-BERT pretraining and several molecular benchmark datasets used in this work. Then, Section 3 presents the molecular representation method, the pretraining, and fine-tuning of the Mol-BERT model, respectively. Moreover, Section 4 analyzes the prediction performance of our proposed method on several molecular datasets and compares it with state-of-the-art sequence-based and graph-based approaches. Finally, the conclusion of this work is summarized in Section 5.

2. Materials

The corpus of chemical compound (i.e., unlabeled SMILES) was obtained from the available ZINC and ChEMBL databases. As a free and available database for virtual screening, ZINC database contains over 230 million purchasable compounds in multiple formats, including ready-to-dock and 3D structures [43]. And ChEMBL database is a manually built database of bioactive molecules with drug-like properties, which collects 1,961,462 distinct compounds [44]. Specifically, we selected compound SMILES from ZINC version 15 and ChEMBL version 27 that can be processed by RDKit software [45], and the duplicates were removed in merged dataset. Moreover, we filtered them by following the same criteria of Mol2Vec [19]. Specifically, the two databases were firstly merged, and duplicates were removed. Then, only compounds SMILES that could be processed by RDKit were kept, and they were filtered according to the following cutoffs and criteria: molecular weight between 12 and 600; heavy-atom count between 3 and 50; clogP21between 5 and 7; and only H, B, C, N, O, F, P, S, Cl, and Br atoms allowed. Additionally, all counterions and solvents were removed, and canonical SMILES representations were generated by RDKit. Finally, this procedure yielded 4 million compounds. Detailed information on the pretraining corpus is provided in Supplementary (available here).

In this paper, we selected 4 widely used benchmark datasets from MoleculeNet [13] to evaluate the performance of our proposed method. SMILES strings were used to encode the input chemical compound in all benchmark datasets. The benchmark datasets we used are introduced as follows:

(i) BBBP. The BBBP dataset provides 2,053 compounds on their permeability properties to predict the barrier permeability

(ii) Tox21. The Tox21 dataset measures 8,014 compounds with their corresponding toxicity data against 12 targets. The label of toxicity is recorded as binary task: if the label value is 1, then it means the compound has toxicity on specific target or 0 otherwise

(iii) SIDER. The SIDER dataset contains a total of 1,427 compounds and their adverse drug reactions (ADR) against 27 system-organ class. The ADR result is described as binary labels

(iv) ClinTox. The ClinTox dataset provides 2 classification tasks for 1,491 drug compounds with known chemical structures, including clinical trial toxicity and FDA approval status

In this paper, we followed the experimental setting of FP2VEC [46], and we split the datasets into the train, validation, and test set with a ratio of 8/1/1. Table 1 shows the detailed description of selected benchmark datasets. Please note that binary and multilabel correspond to the binary and multilabel classification tasks, respectively. And random splitting method randomly splits the samples into training, validation, and test subsets. Scaffold splitting method splits the samples on the basis of their 2D structural frameworks implemented by RDKit software.

3. Methods

In this section, we first describe the overview of our proposed Mol-BERT; then, we separately introduce three modules, which we refer to as the feature extractor, pretraining, and fine-tuning of Mol-BERT, respectively.

3.1. Overview. Figure 1 illustrates the overall process of Mol-BERT. As shown in Figure 1, Mol-BERT consists of three modules, including feature extractor, pretraining, and fine-tuning of Mol-BERT. The Mol-BERT framework learns to predict the molecular property as follows. Given the input drug data (i.e., canonical SMILES), the featurizer module adopts the effective molecular representation to transform them into a set of atom identifier (recall the detail in Feature Extractor). Then, the outputs are fed into a BERT module to obtain a contextual embedding of each molecular substructure through pretraining BERT on vast preprocessed corpus (recall the detail in Pretraining Mol-BERT). Finally, the fine-tuned Mol-BERT outputs a value indicating the probability of certain molecular property in classification task (recall the detail in Fine-Tuning Mol-BERT).

Table 1: The detailed description of selected benchmark datasets.

Dataset	Category	Compound	Tasks	Task type	Split method
BBBP	Physiology	2,053	1	Binary	Scaffold
Tox21	Physiology	8,014	12	Multilabel	Scaffold
SIDER	Physiology	1,427	27	Multilabel	Scaffold
ClinTox	Physiology	1,491	2	Multilabel	Scaffold

3.2. Feature Extractor. The molecular substructure is an important cue for molecular interactions [21, 22]. Therefore, the key idea behind Mol-BERT is that we strengthen to obtain a better representation of molecular substructures by pretraining BERT on the vast unlabeled SMILES sequences. Inspired by Mol2Vec [19] that considered molecular substructures or fragments derived from the Morgan algorithm as "words" and compound as "sentences," here we adopt a similar method to decompose the input SMILES sequences into biological words and sentences.

To achieve it, given an input compound SMILES string, we first obtain its standardize and canonical SMILES representation S generated by RDKit. Then, the Morgan algorithm [11] is used to generate all atom identifiers with radius 0 and 1, denoted by A_i^0 and $A_{i\,i}^1$, respectively, where the subscript i represents the index of each atom. As illustrated in the left part of Figure 1, A_i^0 (i.e., green node) represents the current node set traversed in an atom order while A_i^1 (i.e., Kelly node) represents the neighboring node set connecting directly to the current atom, so A_i^1an be viewed as a kind of substructure or fragment. And A_i are then hashed into a fixed-length vector. Take CC(N)C(=O)O as an example; it consists of six atoms, and we obtain its atom identifiers A_i^0 (i.e., A_1^0-A_6^0) and the corresponding substructures (i.e., A_1^1-A_6^1), and then, they are hashed into a fixed-length vector (e.g., A_1^1 corresponds to 3537119591). Finally, all vectors of the Morgan substructures are summed to obtain the molecular representation. Therefore, in this way, we can generate 119 atom identifiers at radius 0 and 13325 substructure identifiers at radius 1, respectively. The feature extractor module in Mol-BERT can be replaced with various molecular representation methods. For example, FP2Vec [46] can be used as the feature extractor to generate the 1024-bit Morgan (or circular) fingerprint with the predefined radius value.

3.3. Pretraining Mol-BERT. As a contextualized word representation model, BERT [33] adopted the masked technique to predict randomly masked words in a sequence, which can result in learning bidirectional representations. Therefore, Mol-BERT also uses a masked SMILES task (i.e., atom identifier) to predict random substructure in a SMILES string. Different from the traditional way of pretraining language models in NLP that BERT was trained on English Wikipedia and BooksCopus, in this paper, we pretrain Mol-BERT on our preprocessed corpus obtained from ZINC version 15 and ChEMBL version 27 datasets. Specifically, the input SMILES is transformed into a list of atom identifiers A_i

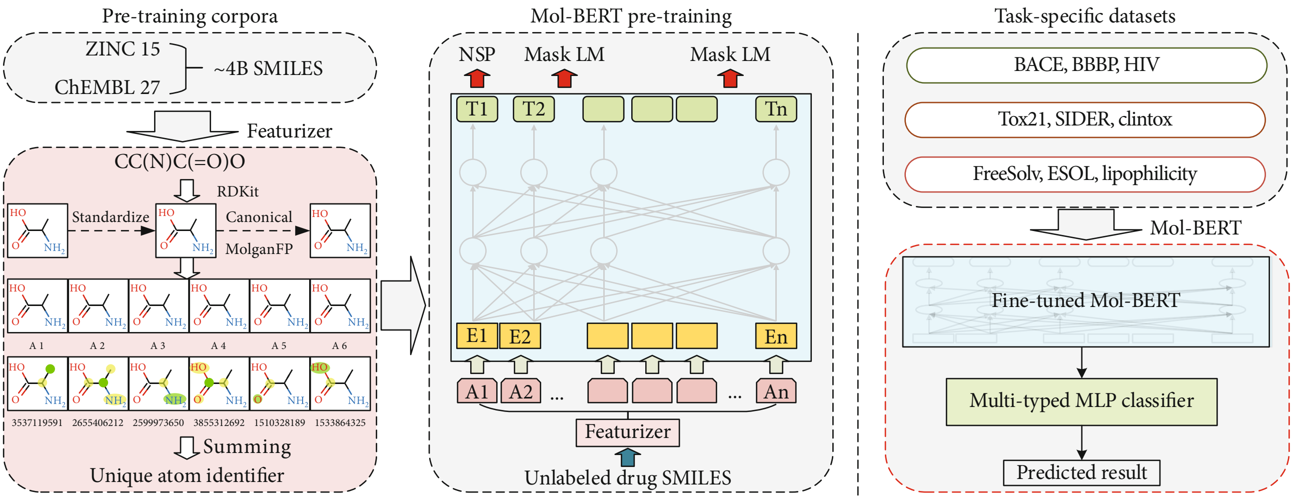


Figure 1: Overview of our proposed Mol-BERT for molecular property prediction.

via a previous module, rather than character-level for SMILES [20], and then, they are embedded as the input of BERT module for pretraining. We initialized our proposed Mol-BERT with weights from BERT [33] and follow the same way to randomly mask 15% tokens in a SMILES (i.e., atom identifier) as [MASK] token. The tokens are embedded into the feature vector. Here, we use token embedding and positional embedding since only the Masked Language Model (MLM) task is adopted in this paper. The proposed Mol-BERT is different from BERT in several ways as follows: (1) Mol-BERT adopted single masked SMILES task (i.e., MLM) on large-scale unlabeled datasets, while BERT uses two kinds of self-supervised tasks on English Wikipedia and BooksCopus, and (2) w exclude the segmentation embedding adopted in the BERT model since Mol-BERT does not require the continuous sentence training.

3.4. Fine-Tuning Mol-BERT. After pretraining on the vast of unlabeled SMILES compounds, with minimal modification of hyperparameters, Mol-BERT can be applied to molecular property prediction on various downstream tasks. We mostly follow the same architecture, optimization, and hyperparameter choices used in [8]. For classification task (i.e., BBBP and Tox21), we feed the final BERT vector into a linear classification layer to predict the molecular property. A simple classifier is adopted to output the binary value. Then, the labeled sample is used for fine-tuning the model. Mol-BERT feeds the learned drug embeddings into a multityped MLP classifier to generate predictions. Output scores include both continuous scores, such as the solubility value and as binary outputs indicating whether a molecule is toxic or nontoxic. The multityped classifier detects whether the task is regression or classification and switches to the correct loss function and evaluation metrics. In the case of regression, we use the mean square error (MSE) as the loss function and root mean square error (RMSE) as performance metrics. In the classification case, we use binary cross entropy as the loss function and area under the receiver operating characteristics (AUC-ROC) as performance metrics. Given a set of SMILES compounds and the ground-truth labels in the training dataset, we used the crossentropy and the mean square error as loss function for classification and regression tasks, respectively.

Table 2: The fine-tuning hyperparameters.

Parameter	Value/range
Learning rate	1*e*-5~1*e*-3
Batch size	8
Epoch	100
Optimizer	Adam
Embedding dimension	300
Size of dictionary	13,325
Number of attention head	6
Layers of fully connected neural network	6

4. Results and Discussion

In this section, we first introduce the experimental settings. Then, we demonstrate the performance of our proposed Mol-BERT in comparison to state-of-the-art methods to predict the molecular property on 4 wildly used benchmark datasets.

4.1. Baseline Methods. We compare Mol-BERT with many state-of-the-art sequence-based and graph-based baselines which can be categorized as follows:

(i) ECFP: extended-connectivity fingerprints, referred to as ECFP [11], are a type of widely used circular or Morgan fingerprints for encoding the substructures in a molecule

(ii) GraphCov: graph convolutions are proposed by [26] to apply the convolutional networks for learning molecular fingerprints. Here, we term it as GraphCov

Table 3: The metric scores of the test set against BBBP, Tox21, SIDER, and ClinTox datasets.

Model/dataset	BBBP	Tox21	SIDER	ClinTox
ECFP	0.702 ± 0.006	0.810 ± 0.013	0.673 ± 0.025	0.783 ± 0.023
GraphCov	0.877 ± 0.036	0.772 ± 0.041	0.593 ± 0.035	0.845 ± 0.051
Weave	0.837 ± 0.065	0.741 ± 0.044	0.543 ± 0.034	0.823 ± 0.023
MPNN	0.913 ± 0.041	0.808 ± 0.024	0.595 ± 0.030	0.879 ± 0.054
FP2VEC	0.874 ± 0.023	0.730 ± 0.006	0.582 ± 0.008	0.643 ± 0.032
SMILES-BERT	0.814 ± 0.093	0.732 ± 0.025	0.601 ± 0.010	0.872 ± 0.017
Mol-BERT	0.875 ± 0.048	0.839 ± 0.075	0.695 ± 0.071	0.923 ± 0.025

(iii) Weave: similar to GraphCov, the weave featurization [27] encodes meaningful features of atom, bond, and graph distances between matching pairs to form molecule-level representations

(iv) MPNN: a novel message passing method is proposed to be operated on undirected graph [30]

(v) FP2VEC: based on Morgan or circular fingerprint, it introduces and encodes a molecule as trainable vectors [46]

(vi) SMILES-BERT: [20] proposes a semisupervised BERT model that takes the SMILES representation as input

We report the results of these baselines in FP2Vec [46], including ECFP, GraphCov, Weave, and FP2VEC. And we reimplemented MPNN and SMILES-BERT, respectively. As for MPNN [30], it is a graph-based model considering the edge features during message passing. And SMILES-BERT [20] is a sequence-based model based on transformer layer and attention mechanisms entirely to encode compound SMILES. These models are relied on the public code and kept the same settings of models the same as reported in the original papers.

4.2. Evaluation Metrics. We applied the area under the receiver operating characteristic curve (AUC-ROC) metric for classification task. Following [46], we train the prediction model with a train set and optimize the model based on the AUC-ROC metric of validation set for classification task. And the prediction results are measured using those optimized models on the test set. For all experiments in this paper, we repeated the same procedures on each task for 5 times and reported the mean and standard deviation of AUC scores. Besides, we evaluated all models on the scaffold splitting method as reported by [46].

4.3. Implementation Details. To optimize all trainable parameters, we adopt Adam optimizer for pretraining and fine-tuning. The dynamic learning rate technique is adopted to adjust the learning rate during training and fine-tuning according to various downstream tasks. We use PyTorch to implement Mol-BERT. And we use 3 NVIDIA GTX 1080Ti GPUs to pretrain Mol-BERT. All fine-tuning tasks are run on a single NVIDIA GTX 2080Ti GPU. Table 2 shows all the hyperparameters of the fine-tuning model.

4.4. Comparison Results. To examine the competitiveness of the proposed model, we compared Mol-BERT with state-of-the-art models used for molecular property prediction on classification task. Table 3 reports the mean and standard deviation of ROC-AUC score on BBBP, SIDER, Tox21, and ClinTox datasets. From this table, we can observe that the proposed Mol-BERT significantly outperforms the baselines across three datasets, including Tox21, SIDER, and ClinTox. More specifically, our proposed Mol-BERT achieved at least 2.9% on Tox21, 2.2% on SIDER, and 4.4% on ClinTox higher ROC-AUC metric than baselines. For example, on the Tox21 dataset, Mol-BERT achieved a ROC-AUC score of 0.839 with 2.9% absolute gain compared to ECFP (the second best method). This is because Mol-BERT leverages the molecular representation pretrained on large- scale unlabeled SMILES sequences, while ECFP heavily relied on feature engineering. Compared with graph-based methods that explore the molecular graph features, the proposed Mol-BERT outperformed them on three datasets while it achieved comparable performance with MPNN on the BBBP dataset. This is due to the fact that the contextual information learned from large unlabeled datasets can benefit a lot to the model performance. Moreover, in comparison to the sequence-based pretrained model (i.e., SMILES-BERT), our proposed Mol-BERT achieved stable performance across all datasets. This is a very encouraging result. The reason could be that our method adopted the molecular representation to consider the structural feature of molecular substructures, which benefits to the performance. Overall, it is essentially a nontrivial achievement in terms of molecular property prediction.

5. Conclusions

In this paper, we proposed an effective molecular representation method with the pretrained BERT model, named Mol-BERT, to resolve the molecular property prediction. Our proposed Mol-BERT leverages the molecular representation of substructures pretrained on large-scale unlabeled SMILES dataset, which is able to learn both structural and the contextual information of drug. We implement the proposed method and conduct experimental comparisons on four widely used benchmarks. The experimental results show that Mol-BERT outperforms the classic and state-of-the-art graph-based models on molecular property prediction.

While our proposed method achieves good performance on classification tasks, there are still some limitations expected to be overcome. First, our method achieves relatively poorer performance on regression task, mainly owing to the small number of samples in the dataset (e.g., FreeSolv). We would like to investigate metalearning strategies for data augmentation, which results in great success in natural language processing. Second, molecular property prediction is the primary step in drug discovery; we will continue to improve our method to further investigate the following prediction task (e.g., protein-protein interaction, drug-disease associations) in the future.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request (https://github.com/cxfjiang/MolBERT).

Conflicts of Interest

The authors declare no competing financial interest.

Supplementary Materials

The pretraining corpus are available at "https://drive.google.com/drive/folders/1ST0WD1-hX9XtiPWwCceZbgZlBV0fKPbe." *(Supplementary Materials)*

References

[1] S. Ekins, A. C. Puhl, K. M. Zorn et al., "Exploiting machine learning for end-to-end drug discovery and development," *Nature Materials*, vol. 18, no. 5, pp. 435–441, 2019.

[2] X. Lin, Z. Quan, Z. J. Wang, H. Huang, and X. Zeng, "A novel molecular representation with BiGRU neural networks for learning atom," *Briefings in Bioinformatics*, vol. 21, no. 6, pp. 2099–2111, 2020.

[3] X. Lin, Z. Quan, Z. J. Wang, T. Ma, and X. Zeng, "KGNN: knowledge graph neural network for drug-drug interaction prediction," in *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, pp. 2739–2745, Yokohama, Japan, 2020.

[4] B. K. Shoichet, "Virtual screening of chemical libraries," *Nature*, vol. 432, no. 7019, pp. 862–865, 2004.

[5] S. Pushpakom, F. Iorio, P. A. Eyers et al., "Drug repurposing: progress, challenges and recommendations," *Nature Reviews Drug Discovery*, vol. 18, no. 1, pp. 41–58, 2019.

[6] Z. Quan, Y. Guo, X. Lin, Z. J. Wang, and X. Zeng, "GraphCPI: graph neural representation learning for compound-protein interaction," in *2019 IEEE International Conference on Bioinformatics and Biomedicine*, pp. 717–722, San Diego, CA, USA, 2019.

[7] Y. Zhou, Y. Hou, J. Shen, Y. Huang, W. Martin, and F. Cheng, "Network-based drug repurposing for novel coronavirus 2019-nCoV/SARS-CoV-2," *Cell Discovery*, vol. 6, no. 1, pp. 1–18, 2020.

[8] D. S. Cao, Q. S. Xu, Q. N. Hu, and Y. Z. Liang, "ChemoPy: freely available python package for computational biology and chemoinformatics," *Bioinformatics*, vol. 29, no. 8, pp. 1092–1094, 2013.

[9] A. Mauri, V. Consonni, M. Pavan, and R. Todeschini, "Dragon software: an easy approach to molecular descriptor calculations," *Match*, vol. 56, no. 2, pp. 237–248, 2006.

[10] H. Moriwaki, Y. S. Tian, N. Kawashita, and T. Takagi, "Mordred: a molecular descriptor calculator," *Journal of Cheminformatics*, vol. 10, no. 1, p. 4, 2018.

[11] D. Rogers and M. Hahn, "Extended-connectivity fingerprints," *Journal of Chemical Information and Modeling*, vol. 50, no. 5, pp. 742–754, 2010.

[12] R. C. Glen, A. Bender, C. H. Arnby, L. Carlsson, S. Boyer, and J. Smith, "Circular fingerprints: flexible molecular descriptors with applications from physical chemistry to ADME," *IDrugs*, vol. 9, no. 3, p. 199, 2006.

[13] Z. Wu, B. Ramsundar, E. N. Feinberg et al., "MoleculeNet: a benchmark for molecular machine learning," *Chemical Science*, vol. 9, no. 2, pp. 513–530, 2018.

[14] 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, United States, 2016.

[15] C. Xia, C. Zhang, X. Yan, Y. Chang, and P. S. Yu, "Zero-shot user intent detection via capsule neural networks," 2018, https://arxiv.org/abs/1809.00385.

[16] J. Yin, C. Gan, K. Zhao, X. Lin, Z. Quan, and Z. J. Wang, "A novel model for imbalanced data classification," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 4, pp. 6680–6687, 2020.

[17] D. Weininger, A. Weininger, and J. L. Weininger, "SMILES. 2. Algorithm for generation of unique smiles notation," *Journal of Chemical Information and Computer Sciences*, vol. 29, no. 2, pp. 97–101, 1989.

[18] Z. Xu, S. Wang, F. Zhu, and J. Huang, "Seq2seq fingerprint: an unsupervised deep molecular embedding for drug discovery," in *Proceedings of the 8th ACM international conference on bioinformatics, computational biology, and health informatics*, pp. 285–294, New York, NY, USA, 2017.

[19] S. Jaeger, S. Fulle, and S. Turk, "Mol2vec: unsupervised machine learning approach with chemical intuition," *Journal of Chemical Information and Modeling*, vol. 58, no. 1, pp. 27–35, 2018.

[20] S. Wang, Y. Guo, Y. Wang, H. Sun, and J. Huang, "SMILES-BERT: large scale unsupervised pre-training for molecular property prediction," in *Proceedings of the 10th ACM International Conference on Bioinformatics, Computational Biology and Health Informatics*, pp. 429–436, New York, NY, USA, 2019.

[21] K. Huang, C. Xiao, T. Hoang, L. Glass, and J. Sun, "Caster: predicting drug interactions with chemical substructure representation," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 1, pp. 702–709, 2020.

[22] R. B. Silverman and M. W. Holladay, *The Organic Chemistry of Drug Design and Drug Action*, Academic Press, 2014.

[23] K. Schütt, P. J. Kindermans, H. E. S. Felix, S. Chmiela, A. Tkatchenko, and K. R. Müller, "SchNet: a continuous-filter convolutional neural network for modeling quantum interactions," *Advances in neural information processing systems*, pp. 991–1001, 2017, https://arxiv.org/abs/1706.08566.

[24] K. T. Schütt, F. Arbabzadah, S. Chmiela, K. R. Müller, and A. Tkatchenko, “Quantum-chemical insights from deep tensor neural networks,” *Nature Communications*, vol. 8, no. 1, pp. 1–8, 2017.

[25] Z. Xiong, D. Wang, X. Liu et al., “Pushing the boundaries of molecular representation for drug discovery with the graph attention mechanism,” *Journal of Medicinal Chemistry*, vol. 63, no. 16, pp. 8749–8760, 2020.

[26] D. K. Duvenaud, D. Maclaurin, J. Iparraguirre et al., “Convolutional networks on graphs for learning molecular fingerprints,” *Advances in Neural Information Processing Systems*, pp. 2224–2232, 2015, https://arxiv.org/abs/1509.09292.

[27] S. Kearnes, K. McCloskey, M. Berndl, V. Pande, and P. Riley, “Molecular graph convolutions: moving beyond fingerprints,” *Journal of Computer-Aided Molecular Design*, vol. 30, no. 8, pp. 595–608, 2016.

[28] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, “Graph attention networks,” 2017, https://arxiv.org/abs/1710.10903.

[29] S. Ryu, J. Lim, S. H. Hong, and W. Y. Kim, “Deeply learning molecular structure-property relationships using attention- and gate-augmented graph convolutional network,” 2018, https://arxiv.org/abs/1805.10988.

[30] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, *Neural Message Passing for Quantum Chemistry. in International Conference on Machine Learning*, PMLR, 2017.

[31] K. Yang, K. Swanson, W. Jin et al., “*Are learned molecular representations ready for prime time?, [Ph.D. thesis]*,” Massachusetts Institute of Technology, 2019.

[32] Y. Song, S. Zheng, Z. Niu, Z. H. Fu, Y. Lu, and Y. Yang, “Communicative representation learning on attributed molecular graphs,” in *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, pp. 2831–2838, Yokohama, Japan, 2020.

[33] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, “BERT: pre-training of deep bidirectional transformers for language understanding,” in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*, vol. 1, pp. 4171–4186, Minneapolis, United States, 2019.

[34] W. Hu, B. Liu, J. Gomes et al., “Strategies for pre-training graph neural networks,” 2019, https://arxiv.org/abs/1905.12265.

[35] K. Li, Y. Zhong, X. Lin, and Z. Quan, “Predicting the disease risk of protein mutation sequences with pre-training model,” *Frontiers in Genetics*, vol. 11, p. 1535, 2020.

[36] B. Song, Z. Li, X. Lin, J. Wang, T. Wang, and X. Fu, “Pretraining model for biological sequence data,” *Briefings in Functional Genomics*, vol. 20, no. 3, pp. 181–195, 2021.

[37] A. Vaswani, N. Shazeer, N. Parmar et al., “Attention is all you need,” in *Advances in Neural Information Processing Systems*, NIPS (Conference and Workshop on Neural Information Processing Systems), 2017.

[38] S. Min, S. Park, S. Kim, H. S. Choi, and S. Yoon, “Pre-training of deep bidirectional protein sequence representations with structural information,” 2019, https://arxiv.org/abs/1912.05625.

[39] R. Rao, N. Bhattacharya, N. Thomas et al., “Evaluating protein transfer learning with tape,” in *Advances in Neural Information Processing Systems*, NIPS (Conference and Workshop on Neural Information Processing Systems), 2019.

[40] K. Huang, J. Altosaar, and R. Ranganath, “ClinicalBERT: modeling clinical notes and predicting hospital readmission,” 2019, https://arxiv.org/abs/1904.05342.

[41] 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.

[42] P. Schwaller, T. Laino, T. Gaudin et al., “Molecular transformer: a model for uncertainty-calibrated chemical reaction prediction,” *ACS Central Science*, vol. 5, no. 9, pp. 1572–1583, 2019.

[43] J. J. Irwin, T. Sterling, M. M. Mysinger, E. S. Bolstad, and R. G. Coleman, “ZINC: a free tool to discover chemistry for biology,” *Journal of Chemical Information and Modeling*, vol. 52, no. 7, pp. 1757–1768, 2012.

[44] D. Mendez, A. Gaulton, A. P. Bento et al., “ChEMBL: towards direct deposition of bioassay data,” *Nucleic Acids Research*, vol. 47, no. D1, pp. D930–D940, 2019.

[45] J. Woosung and K. Dongsup, *RDKit: Open-Source Cheminformatics*, 2006, https://www.rdkit.org.

[46] W. Jeon and D. Kim, “FP2VEC: a new molecular featurizer for learning molecular properties,” *Bioinformatics*, vol. 35, no. 23, pp. 4979–4985, 2019.

Hindawi
Wireless Communications and Mobile Computing
Volume 2021, Article ID 1391801, 14 pages
https://doi.org/10.1155/2021/1391801

Research Article

Joint Generative Image Deblurring Aided by Edge Attention Prior and Dynamic Kernel Selection

Zhichao Zhang,[1] **Hui Chen**,[2] **Xiaoqing Yin**,[3] **and Jinsheng Deng**[3]

[1]*College of Computer, National University of Defense Technology, Changsha 410000, China*
[2]*Science and Technology on Integrated Logistics Support Laboratory, National University of Defense Technology, Changsha 410000, China*
[3]*College of Advanced Interdisciplinary Studies, National University of Defense Technology, Changsha 410000, China*

Correspondence should be addressed to Zhichao Zhang; 1933978660@qq.com

Received 27 May 2021; Accepted 2 July 2021; Published 1 August 2021

Academic Editor: Jerry Chun-Wei Lin

Image deblurring is a classic and important problem in industrial fields, such as aviation photo restoration, object recognition in robotics, and autonomous vehicles. Blurry images in real-world scenarios consist of mixed blurring types, such as a natural motion blurring owing to shaking of the camera. Fast deblurring does not deblur the entire image because it is not the best option. Considering the computational costs, it is also better to have an alternative kernel to deblur different objects at a high-semantic level. To achieve better image restoration quality, it is also beneficial to combine the blurring category location and important structural information in terms of specific artifacts and degree of blurring. The goal of blind image deblurring is to restore sharpness from the unknown blurring kernel of an image. Recent deblurring methods tend to reconstruct prior knowledge, neglecting the influence of blur estimation and visual fidelity on image details and structure. Generative adversarial networks(GANs) have recently been attracting considerable attention from both academia and industry because GAN can perfectly generate new data with the same statistics as the training set. Therefore, this study proposes a generative neural architecture and an edge attention algorithm developed to restore vivid multimedia patches. Joint edge generation and image restoration techniques are designed to solve the low-level multimedia retrieval. This multipath refinement fusion network (MRFNet) can not only perform deblurring of images directly but also individual the frames separately from videos. Ablation experiments validate that our generative adversarial network MRFNet performs better in joint training than in multimodel. Compared to other GAN methods, our two-phase method exhibited state-of-the-art performance in terms of speed and accuracy as well as has a significant visual improvement.

1. Introduction

GANs have exhibited a promising performance on edge restoration and image deblurring [1, 2]tasks. However, restoration methods typically introduces artifacts if the blurred area has uniform intensity, because it selects an incorrect region for deblurring. Deep learning approaches have been proposed to handle complex natural blurring. These methods use convolutional layers to extract features by scanning blurred and sharp images and subsequently fusing features with deconvolution layers and recording the learned results [3–5]. Xu et al. [6], Schuler et al. [7], and Zhang et al. [8] adopted this two-stage traditional procedure based on the use of an encoder-decoder neural network. However, these methods still adopt the traditional framework with low prediction performance.

Inspired by the problems described above, Kupyn et al. [9] designed a new framework for deblurring that could calculate the differences between generative and original images. GANs have shown promising performance in image deblurring. Scholars have also achieved significant improvements using other complicated GAN networks, such as DeblurGAN [9], DeblurGANv2 [10], and EGAN [1, 2, 11, 12]. However, a GAN requires a large amount of computational and memory resources when comparing the generated and original images of the discriminator. With advancements in the design of complicated network models, more complex end-to-end deep learning approaches have been proposed for deblurring. These networks can be divided into four classes: including multiscale, recurrent, multipatch, and scale-iterative networks.

The frameworks of Nah et al. [13] and Lin et al. [14] employ a multiscale style. The main idea of their frameworks is the implementation of the coarse-to-fine strategy to deblur images in consecutive stages. The coarse stage obtains features by using scales, and the features are then halved in a series of steps. The fine stage learns the larger-scale features with the aid of the coarse features until the original size is reached. The coarse-to-fine mechanism is performed directly via the scale-cascaded structure. However, despite the achievement of suitable results, such networks size and depth eventually become excessive, thus leading to increased graphics processing unit (GPU) memory consumption.

Tao et al. proposed the recurrent architecture in which subsequentthe next rounds of training can be aided by the results of the previous round [15]. Multipatch networks have been proposed by Nekrasov et al. [16] and Zhang et al. [17], whereby the recurrent method was applied by regarding the last-turn results as the next round input for refining final checkpoints. Images are separated into patches and extracted features, and the meaningful results are sent to the next iteration for further enhancement. This method can be conducive to the reduction of the parameters by learning from patches in a single round. However, the method is not stable for a complex blurring.

Ye et al. [18] used the scale-iterative architecture to train the model by applying an upsampling path with the aid of the results of the previous iteration, blurring kernels varied in different regions. Low-frequency information was present, such as semantic and category contents, and background color, along with high-frequency information, such as edge and structure. High-spatial gradients are diminished more in blurred or low-resolution images. Hence, we combine the ideas of multiscale and recurrent architectures to produce a new framework. The design of the MRF network overcomes the parameter and low-efficiency issues of multiscale and recurrent architectures, respectively. Considering the above limitations and strengths, we propose a multipath refinement network called MRFNet. The main contributions of this study are summarized as follows.

Firstly, in terms of the network, we develop a multipath refinement network (MRFNet) for joint low-level image training, with a plug-and-play feature for multiple attention modules. It is plug and play for several edge detection networks for image information prior and feature extraction, and multiple attention modules can also be added at multiscale dataflow paths. An iterative and recurrent strategy is first designed to train a lightweight yet efficient network. We design a deblurring network to search the blurring kernels dynamically, fully exploiting the attention mechanism to focus on the blurring area.

Secondly, universally, image restoration of edge attention is preformed in three steps. First, we abstract the edge information by edge prior, the proposed approach refines the inside features by attention modules to finally reconstruct the whole image. Reconsidering edge attention mechanism for the image prior, we develop a general algorithm for low level image restoration. This method applies a different feature extraction sequence: objects are targeted by a class activated function. Only the main structures and key features of the marked object can be recognized by edge detection. Finally, preset proper kernels are adopted to process the suitable regions.

Thirdly, several techniques are investigated ablation experiments to explore various deep learning strategies. In this study, we verify that image deblurring performs better in joint training than transfer learning or multimodel training. An edge attention algorithm, lightweight residual strategy, fine-tuned weight, and multipath refinement loss function are developed in a plug-and-play architecture to adapt different demands for image processing efficiency, GPU requirements of the model, speed and accuracy balance, and training efficiency. We modify the network in a lightweight manner by combining the iterative and recurrent architectures. The design of a lightweight convolution and residual connection network architecture makes the model more streamlined, efficient, and fast.

The remainder of this study is organized as follows. We introduce the related work on image deblurring in network architectures in Section 2. Section 3 illustrates the methodology and outlines the implementation of our proposed network. We discuss our experimental results in Section 4 and present our conclusions in Section 5.

2. Related Work

2.1. Blurring Kernel Estimation. The early work on image deblurring depended on a variety of assumptions and natural images acquired a priori [19]. Subsequently, some uncertain parameters would be determined in blurring models, such as the type of blurring kernel and additive noise [20, 21]. However, in real applications, these simplified assumptions about sampled scenes and blurring models may lead to performance degradation. In addition, these methods are computationally expensive, and numerous parameters typically need to be adjusted.

In recent years, the application of deep learning and generative networks in computer vision tasks has led to great breakthroughs in many research fields. Several regression networks based on convolution neural network (CNNs) have been proposed for image restoration, including some methods that involve with image deblurring [22, 23]. Compared with traditional methods, the methods based on deep learning are less dependent on prior knowledge. These new models have demonstrated the ablity to reconstruct images more accurately on both global and local scales.

It is generally believed that a blurred image is formed by the convolution of a blurring kernel and additive noise [3]. Therefore, the existing algorithms normally use the blurring kernel function for the deconvolution of a blurred image. The existing algorithms can be divided into two categories, according to whether the blurring kernel is known, including (a) blind image deconvolution (BID) [16, 24, 25] and (b) nonblinded image deconvolution (NBID) [20, 22]. BID restores a clear image without knowledge of a blurred kernel. It only knows the blurred images it has captured. NBID deblurs images with a known blurring kernel. It is usually difficult to know the blurring kernel in practical applications in advance. Therefore, the requirements of BID are much higher than those of NBID.

Such models may use a known fixed kernel to blur [20, 26]. Recent studies have used end-to-end learning methods to handle the blurring of spatial changes, achieving state-of-the-art performance [23, 27].

Some problems remain with prior deep neural network architecture for image deblurring. First, although neural networks that use deeper architectures are usually effective, it is difficult to explain the impact of individual components in these networks. Second, the evaluation indicators used in image restoration tasks, such as PSNR and SSIM, are usually based on pixel or feature differences between clear natural images and processed images. This tends to improve mathematical similarity rather than the quality of human subjective perception. PSNR measures image quality by calculating the mean square error (MSE). However, there is a gap between the MSE and evaluation performed by a human visual system. SSIM models human visual quality in terms of multiple components, such as brightness, contrast, and structure. These components can be used to assess visual quality, but they are essentially unilateral assessments of the complexity of human vision.

On the assumption of fixed blurring kernels for sensors, we can consider it as a mean blurring operation and can use it to model the blurring estimation as a convolution of a latent image I and blurring kernel k,

$$B = k * I + \alpha, \tag{1}$$

where B and α represent the blurring image and added noise, respectively, and "$*$" is the convolution operator. This is a mathematically ill-posed problem, because different I and k pairs can produce the same B values.

2.2. Attention Mechanism Screening Blurring Kernel. In this study, we reviewed the global average pooling layer proposed in [5] and illustrate how it explicitly enables CNNs to have excellent location capabilities, despite training on image-level tags. Although this technique has been previously proposed as a method for regularization training, we find that it establishes a universally localizable deep representation that can be applied to a variety of tasks. We locate objects with high accuracy even though the global average pool appears simple. Furthermore, we demonstrate that our network can locate differentiated image regions for a variety of tasks, even without training.

The latest work of Zhou et al. [28] shows that the convolution units of each layer of the CNN act as object detectors for the location of objects, even without supervision. This function is lost when classifying objects with fully connected layers. Popular CNNs have recently been proposed to avoid the use of fully connected layers to minimize the number of parameters, while maintaining high performance [5]. To achieve this goal, Lin et al. [5] used global average pooling (GAP) as the structure regulator to prevent overfitting.

It is important to highlight the intuitive difference between GAP and global maximum pooling (GMP). GMP encourages the identification of only one discriminatory part, while GAP encourages the network to identify a range of objects. It is designed to replace fully connected layers in classical CNNs. GMP has been used for weakly supervised object locations in previous research [29]. In our experiments, we found that the advantages of GAP layers extended beyond their functionality as a normalization regulator. With a small adjustment, the network can retain its excellent localization capabilities to the last layer. Distinguishable image areas can be easily identified in a single forward pass using this adjustment to accomplish a variety of tasks, even those for which the network was not initially trained.

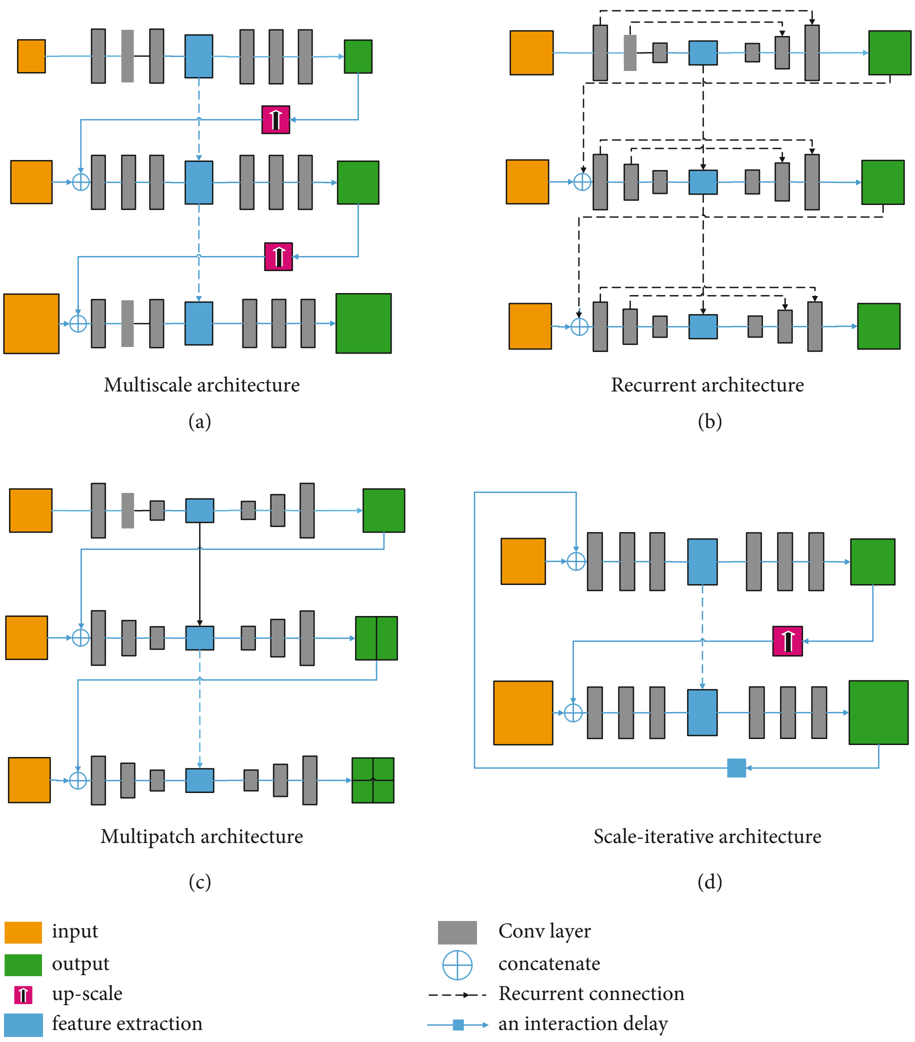
The aim is for each unit to be activated by a visual pattern in its receptive area. Therefore, a map of the visual mode is required. The class activation graph is the weighted linear sum of the presence of these visual patterns in different spatial locations. The most relevant images areas to a particular category can be identified by simply sampling the class activation graph to the size of the input image.

Traditional methods rely on blur kernel estimation to reconstruct images by focusing on specific types of blurs [3, 24, 30–32]. Recent studies have attempted to settle the restoration problem by adopting multiscale CNNs to deblur the images. In these end-to-end frameworks, blurry images are used as inputs to the neural network to immediately generate clear images [20]. However, the performance of such methods remains unsatisfactory owing to the fixed assumption of the blurring kernel. CNNs can greatly improve the computational speed of traditional methods, but their prediction accuracy remains inefficient, and they require the use of considerable GPU memory resources.

2.3. Network Architecture. Image deblurring CNNs can be divided into GAN, multiscale, recurrent, multipatch, and scale-iterative architecture networks for feature extraction.

2.3.1. Multiscale Architecture. Multiscale networks [13] extract various features from each scale by scaling an image into different sizes, as shown in Figure 1(a). The input images are converted into feature maps, and scales are used to halve the feature maps at each level. In multiscale detection, the various scale features are fused with different methods and contain a large quantity of information, suggesting the possibility of high accuracy. However, the multiscale strategy strictly requires the features to be extracted from the small scale to the large scale; which means that large-scale concatenation processes must wait for the computational results from the small scales, which results in a relatively slow training speed.

2.3.2. Recurrent Architecture. An input layer, loop hiding layer, and output layer constitute a recurrent network [18, 33, 34] as shown in Figure 1(b). Recurrent networks can learn features and long-term dependencies in sequence. However, as the number of network layers increases, so does the


Multiscale architecture

(a)

Recurrent architecture

(b)

Multipatch architecture

(c)

Scale-iterative architecture

(d)

Figure 1: Various deblurring network architectures. (a) Nah et al. [13] proposed the multiscale architecture to extract features from different scales. (b) Tao et al. [15] proposed the recurrent architecture, in which the next round of training can be aided by the last round results. (c) Zhang et al. [17] utilized the multipatch architecture to directly extract features from image pairs by cropping images in different scales. (d) Ye et al. [18] used the scale-iterative architecture to train the model with an upsampling path with aid of the last-iterative middle results. We combine the ideas of (a) and (b) and propose a new framework whose core module involves the MRF and call it MRFNet. The MRFNet can operate in both multiscale and recurrent manner.

required computational complexity. The process deteriorates if invalid features are extracted in the last round because the concatenation of recurrent networks relies heavily on last-round results. Subsequently, the deblurring inference becomes extremely unstable if image restorations are of poor quality.

2.3.3. Multipatch Architecture. A deep multipatch hierarchical network (DMPHN) is a CNN model that appears simple but operates as an effective multipatch network, as shown in Figure 1(c) [17]. An input image is divided into different sizes each time. Features were then extracted with the use of a multiscale architecture. Although DMPHN has attained remarkable progress in terms of computational effectiveness, their precision is low.

2.3.4. Iterative Architecture. Ye et al. [18] proposed a scale-iterative upscaling network (SIUN) to iteratively restore sharp images, as shown in Figure 1(d). The super-resolution structure of an upsampling layer was adopted between two consecutive scales to restore the details. Image features are extracted from small to large scales, with the aim of reconstructing high-resolution images from low-resolution originals. The downsampling process begins to restore the image until it is equal to the size of the original image. Moreover, its weight sharing can be preserved, and its training process is flexible. However, the method failed to achieve high deblurring precision and network efficiency, and substantial memory was required for the iterative calculation.

We extend this method by combining the edge feature learning strategy and contextual attention modules for further image restoration, which can locate objects aided by structure information and adopt appropriate deblurring priors to reconstruct sharp images.

3. Model Design and Implementation

The MRFNet is extensively constructed to ensure a balance between accuracy and speed. We first exploit the recurrent and multiscale strategies to learn multifrequency information. A structure is designed with a branch depth and fusion unit on basis of the lightweight process and remote residual connection [35]. Finally, a multiscale refinement loss function is used to train the network in a coarse-to-fine manner.

3.1. Multiscale and Recurrent Learning. The recurrent and multiscale learning strategies are applied in this study. The basic idea of the multiscale learning strategy is to extract features from large coarse scale maps and upsampled results as green lines shown in Figure 2(a). Meanwhile, in the recurrent learning strategy, the high-level feature extraction path acquires fusion information from the low-level refinement maps and the final feedback in the form of purple flow lines, as shown in Figure 2(a). In our study, the two strategies are combined by designing four refinement paths to extract features in different scales, instead of directly predicting the entire deblurred image. Thus, the network only needs to focus on learning highly nonlinear residual features, which is effective in restoring deblurred images in a coarse-to-fine manner. The architecture of the proposed MRFNet is shown in Figure 2.

In the multipath input stream illustrated in Figure 2(a), the upper MRFNet layer takes blurred and sharp images as input and processes the deblurring datasets in a total of four scales, i.e., k varies from 2 to 4. The four scale blurring feature maps are denoted as b_k, while the refinement results are denoted as l_k. First, the k level of the multipath input stream concatenates the same scale feature maps b_k and upsampling feature maps l_{k+1} into a middle feature map denoted as

$$c_k = b_k \oplus l_{k+1} (2 \leq k \leq 4). \tag{2}$$

The fusion unit then adds c_k and the results from the last iteration l_{k-1} to obtain the final outcomes, which is denoted as l_k. This process briefly describes how the refinement fusion path functions. The entire process can be calculated as

$$l_k = c_k + l_{k-1} (2 \leq k \leq 4). \tag{3}$$

3.2. Lightweight Residual Process. Numbers of parameters and floating-point operations of our original MRF network originate from the commonly used 3 × 3 convolution. Therefore, we focus on the replacement of these elements with simpler counterparts without compromising performance.

The original design of our MRFNet employs an encoder–decoder structure equipped with four feature extraction and downsampling layers. Each path includes a fusion unit. The basic block uses a 3 × 3 convolution, which we call the fusion unit. Herein, the 1 × 1 fusion unit in Figure 3(a) is replaced with a 3 × 3 convolution. A chained residual pool (CRP) is also considered to naturally illustrate the operation of the lightweight process and how the three former units are reshaped. The lightweight process is applied to the CRP unit by substituting the 5 × 5 and 3 × 3 convolutions with the 5 × 5 and 1 × 1 convolutions, respectively, as shown in Figure 3(b).

The refinement path adopts a convolution layer with a stride of 1 followed by a convolution layer with a stride of 2, such that they consistently shrink the feature map size by half. The two convolution layers act as a residual connection unit (RCU). Two RCUs are installed in the encoder, and three RCUs are installed in the decoder. All blocks use 1 × 1, 3 × 3, and 1 × 1 convolutions compared with those in the RCU that use 3 × 3 and 3 × 3 convolutions. We call the two convolution layers the lightweight residual connection unit (LWRCU), as illustrated in Figure 3(c).

Intuitively, a convolution with a relatively large core size is designed to increase the size of the receiving field as well as the global context coverage. The 1 × 1 convolution can only transform the features of each pixel locally from one space to another. Herein, we empirically prove that the replacement with a 1 × 1 convolution does not weaken the network performance. Specifically, we replaced the 3 × 3 convolutions in the CRP and fusion block with a 1 × 1 counterpart. We also modify the RCU to LWRCU with a bottleneck design, as shown in Figure 3(c). This method was able to reduce the number of parameters by more than 50% and the number of triggers by more than 75%, as shown in Table 1. The convolutions have been shown to save considerable computation time without sacrificing performance.

We also enhanced the MRF unit, as illustrated in Figure 3(d). Deep residual networks obtain rich feature information from multisize inputs [36]. Residual blocks, originally derived for image classification tasks,are extensively used to learn robust features and train deeper networks. Residual blocks can address vanishing gradient problems. Thus, we replaced the connection layer in the MRF unit.

Herein, the MRF is specifically designed as a combination of multiple convolution layers (conv-f-1 to conv-f-5), and each convolution layer is followed by a rectifier linear unit activation function. Conv-f-2 uses feature maps generated by conv-f-1 to generate more complex feature maps. Similarly, conv-f-4 and conv-f-5 continue to use the feature map generated by conv-f-3 for further processing. Finally, the feature maps obtained from multiple paths are fused together. The specific calculation expression is given as follows:

$$y = f_2(f_1(x)) + f_4(f_3(f_2(f_1(x)))), \tag{4}$$

where f, x, and y represent the convolution operation, characteristic graph of the input, and characteristic graph of the output, respectively.

We construct a residual connection in each path of the MRFNet. In the process of forward transmission, the remote residual connections transmit low-level features, which are used to refine the visual details of coarse high-level feature maps. The residual connections allow the gradients to propagate directly to the early convolution layers, thus contributing to effective end-to-end training.

We set the number of paths from 1 to 6 for the multipath process. The operation used the least number of parameters when the number of paths is 3, whereas better performance is achieved when the number of paths was 4. When the number of paths was less than 3, the extracted features were in accurate. When the number of paths exceeds 4, the

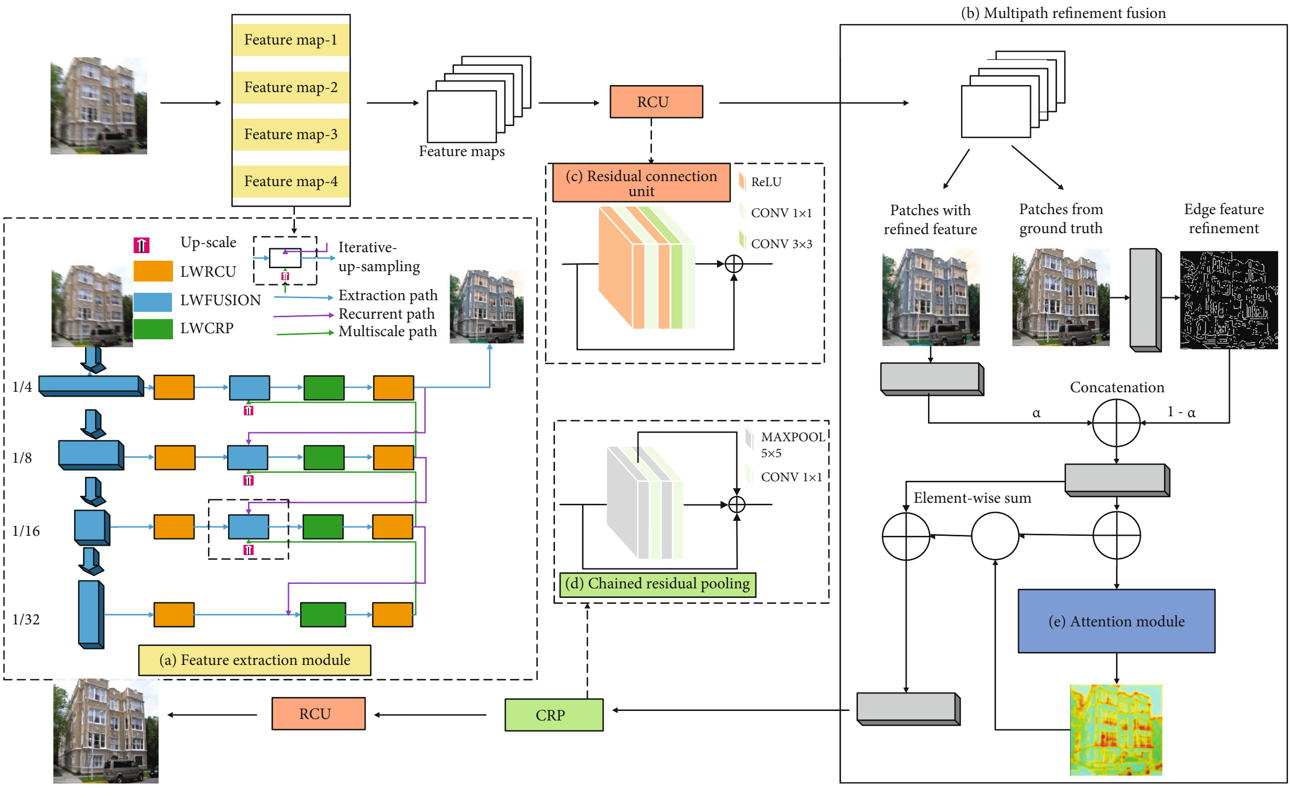


Figure 2: MRF framework. The image is separated into different scales from top to bottom. (a) The extraction path of extracting features from scales. (b) Fusion of the recurrent last-round results and the upsampling feature maps as a single refinement process. All four refinement paths finally compute the loss in the scale refinement loss function, and then, the best deblur results are obtained.

deblurring process encountered severe performance degradation, and the training loss remains at a high level continuously. To this end, we chose the four-path refinement setting as the final backbone.

3.3. Loss Design and Training Strategy. Given a pair of sharp and blurred images, MRFNet takes them as input and produces four groups of feature maps at different scales. The input image size is $H \times W$. The four scales of the feature maps are $H/4 \times W/4$, $H/8 \times W/8$, $H/16 \times W/16$, and $H/32 \times W/32$. Loss design: in the training process, we adopt an L2 loss between the predicted deblurring result map and the ground truth, as follows:

$$L(\theta) = \frac{1}{2N}\sum_{i=1}^{N}\left\|x_s^i - F\left(x_l^i\right)\right\|^2, \qquad (5)$$

where θ is the parameter set, x_i is the ground truth patch, and F is the mapping function generating the restored image from the N-interpolated LR training patches x_l. Herein, the patch size is defined at different levels.

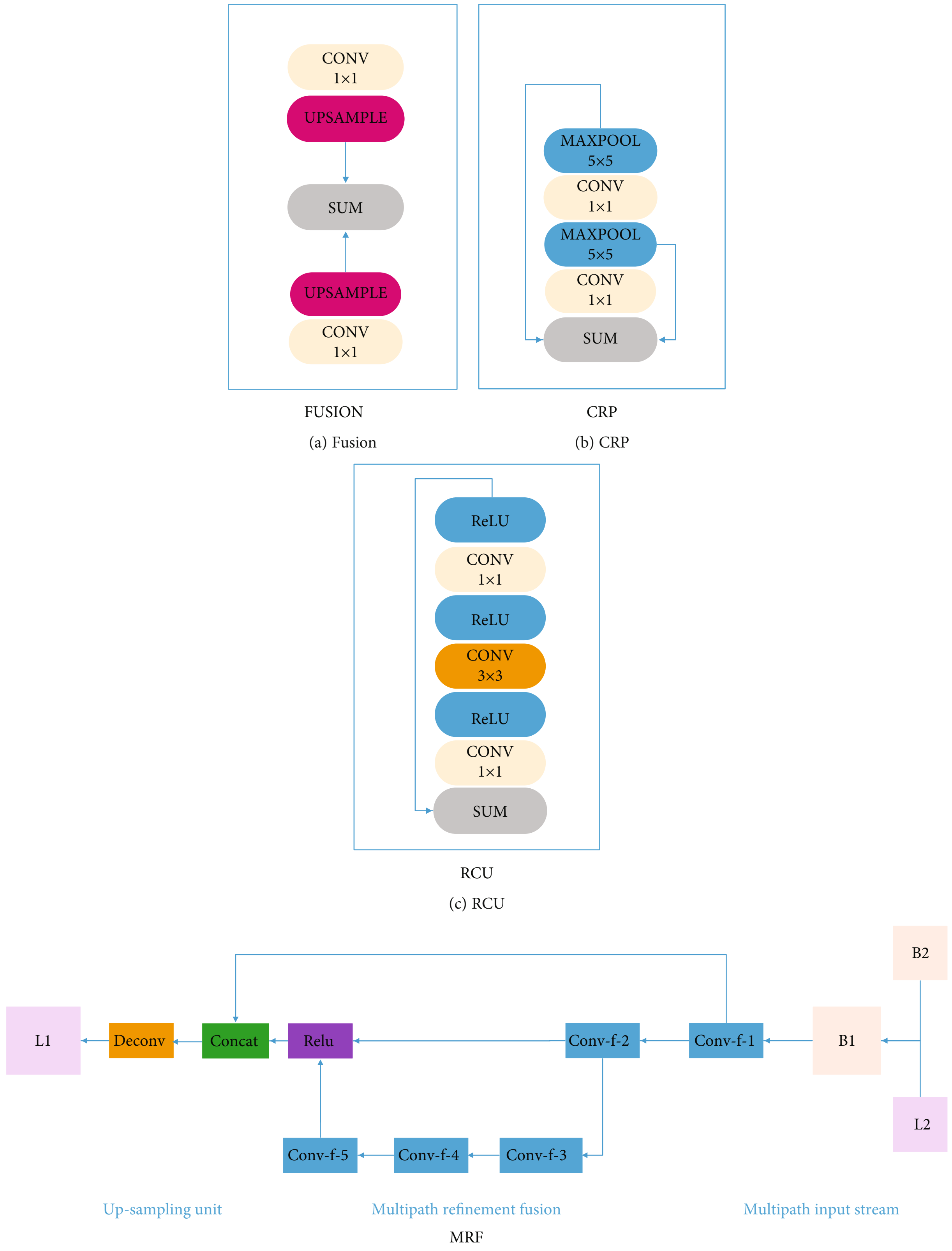
The multiscale refinement loss function is useful in learning the features in a coarse-to-fine manner. Each refinement path includes a loss function that can be used to evaluate the training process. Moreover, our scale refinement loss function computes the results at different scales, which leads to a much faster convergence speed and an even higher inference precision. The final loss is calculated as follows:

$$L_{\text{final}} = \frac{1}{2K}\sum_{k=1}^{K}\frac{1}{c_k w_k h_k}\left\|L_k - S_k\right\|^2 + L_{\text{edge}}, \qquad (6)$$

where L_k represents the model output of the scale level K and S_k denotes the k-scale sharp maps. The loss at each scale is normalized by the number of channels c_k, width w_k, and height h_k.

Progressive weighted training process: the entire feature extraction and fusion process is illustrated in Figure 2(b). In the multipath refinement extraction and fusion stages, the task is to fuse the deblurring feature and edge feature from the outputs to generate the final restored frame. The patches with blurry and refined features and the ground truth are input during the training process.

First, the edge feature is extracted from the ground truth patches and the hyper parameter α is initially set to 0 to control the proportion of the refined resource. Second, the refined and mixed edge feature patches are fused in the contextual attention module, which uses the softmax function to predict the foreground and generate the preliminary activated heatmaps. Third, α is set to 1, and the deblurred, refined feature patches are sent to the attention module in the middle of the training process and are then predicted again by the attention module. The results are compared with the synthesis loss function between the predicted deblurring



(a) Fusion

(b) CRP

(c) RCU

(d) MRF

Figure 3: Details of convolutionxal layers: (a) fusion unit, (b) improved CRP module, (c) lightweight network structure of RCU, and (d) MRF unit.

Table 1: Specific parameters of the MRFNet.

Network	Kernel	Stride	Padding	Network	Kernel	Stride	Padding
Conv1	5 × 5 × 32	1	2	conv_r2_m2	1 × 1 × 128	1	1
Conv2	1 × 1 × 64	1	1	conv_r2_m3	3 × 3 × 128	1	1
Conv3	5 × 5 × 128	2	2	conv_r2_m4	1 × 1 × 128	1	1
Conv4	1 × 1 × 128	1	1	deconv2	4 × 4 × 64	1	2
Conv5	3 × 3 × 256	1	2	conv_r3_1	3 × 3 × 64	1	1
Conv6	1 × 1 × 256	1	1	conv_r3_m1	3 × 3 × 64	1	1
Conv7	3 × 3 × 256	1	2	conv_r3_m2	1 × 1 × 64	1	1
Conv8	1 × 1 × 256	1	1	conv_r3_m3	3 × 3 × 64	1	1
conv_r1_1	3 × 3 × 256	1	1	conv_r3_m4	1 × 1 × 64	1	1
conv_r1_m1	3 × 3 × 256	1	1	deconv3	4 × 4 × 32	1	2
conv_r1_m2	3 × 1 × 256	1	1	conv_r4_1	3 × 3 × 32	1	1
conv_r1_m3	3 × 3 × 256	1	1	conv_r4_m1	3 × 3 × 32	1	1
conv_r1_m4	3 × 1 × 256	1	1	conv_r4_m2	3 × 3 × 32	1	1
deconv1	4 × 4 × 128	1	2	conv_r4_m3	1 × 1 × 32	1	1
conv_r2_1	3 × 3 × 128	1	1	conv_r4_m4	3 × 3 × 32	1	1

results and patches with sharp features. Therefore, the deblurring feature refines the input of blurry images and benefits the edge feature extraction at the beginning of the training. In the middle of the training process, the deblurring and edge features are fused by controlling the parameter α. Finally, each path containing different scales of double feature patches is refined and matched with the use of the multipath context attention module with activated heatmaps to infer the final predictions.

4. Performance Evaluation

In this section, we compare MRFNet to recently adopted methods specifically, DeepDeblur [37], DeblurGAN [9], DeblurGANv2 [10], DMPHN [17], and SIUN [18], in terms of accuracy and time efficiency.

4.1. Experimental Setup. MRFNet was implemented using the Caffe deep learning framework. The model was trained with the Adam optimizer ($\beta_1 = 0.9$, $\beta_2 = 0.999$). Input images were randomly cropped to 256 × 256 in the training process. A batch size of 16 was used for the training, which are performed with four NVIDIA RTX2080Ti graphical processing GPUs. At the beginning of each epoch, the learning rate was initialized to 10^{-4} and was subsequently halved every 10 epochs. We trained for 170 epochs on the VisDrone dataset and 150 epochs on GOPRO.

For the sake of time efficiency, we evaluated the inference time of the existing state-of-the-art CNNs on an 11 GB RTX2080Ti GPUs.

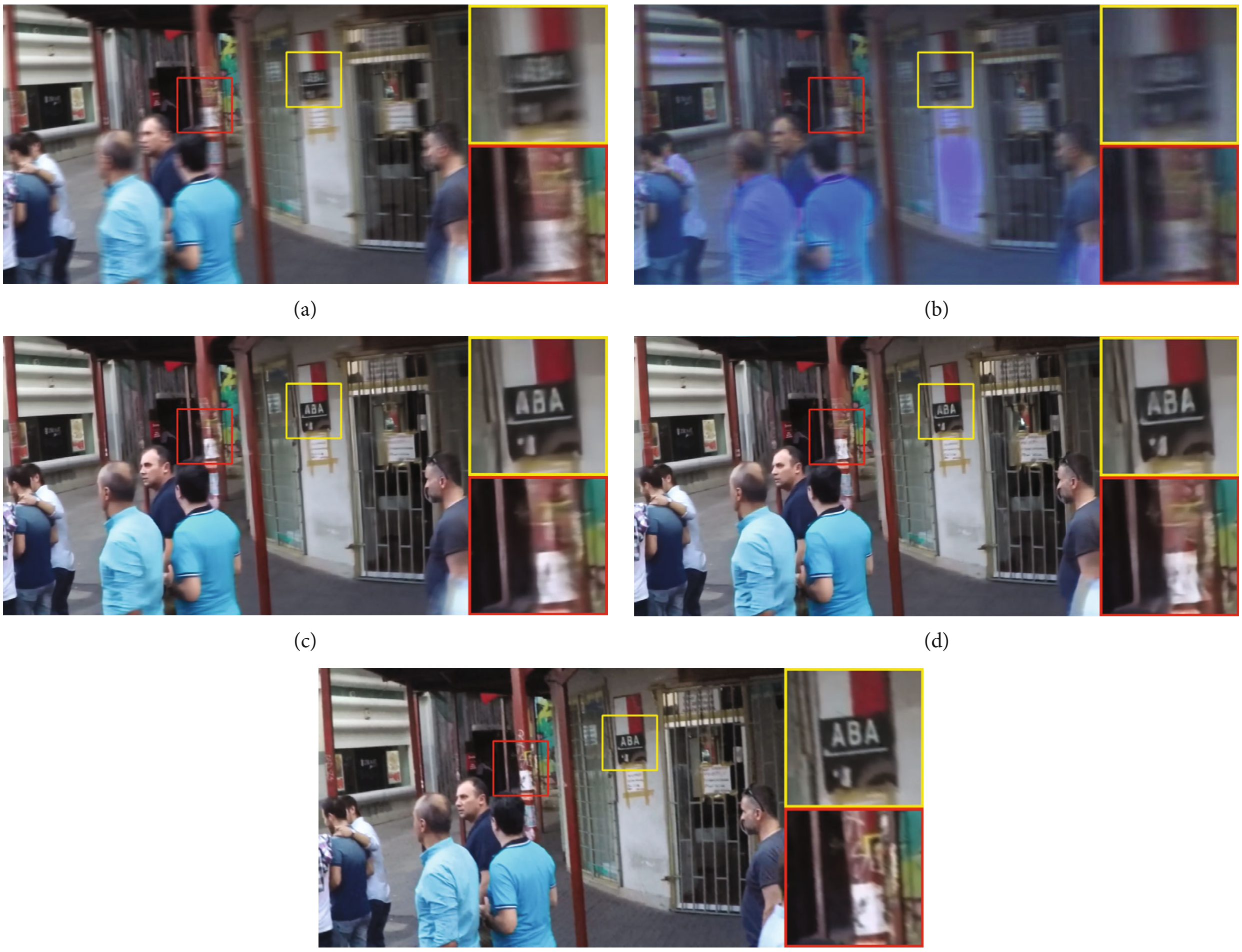
4.2. Dataset. We used two popular benchmark datasets to train and evaluate the performance of MRFNet: VisDrone and GOPRO. VisDrone provides synthetic blurring techniques and collects real blurry aerial scenarios [38]. GOPRO captures real-world motion blurring scenarios [9]. The images collected from GOPRO were 1280 × 768, while those of VisDrone were 256 × 256. The VisDrone dataset included extreme blurry and distorted texture augmentation.

4.3. Comparative Experiments. We conducted comparative experiments using on DeepDeblur [37], DeblurGAN [9], DeblurGANv2 [10], DMPHN [17], and SIUN [18] to verify the performance of our proposed model. The visual effects of different methods are illustrated in Figure 4. MRFNet achieved state-of-the-art performance compared with SIUN and demonstrated clear object boundaries without artifacts Figure 5. The PSNR and SSIM values for MRFNet were much higher than those for DeblurGAN, DeepDeblur, and DMPHN.

Moreover, our method performed better than SIUN and DMPHN and much better than DeblurGANv2 in addressing the GOPRO motion blurs. The trends in Table 2 prove the superiority of the MRFNet framework based on the PSNR and SSIM values. Other methods show considerable limitations in SSIM, indicating that they lack the capacity to restore missing significant structural information and perform deblurring on extremely blurry images.

$$\begin{aligned}
\text{claim} &: n_{\text{MRF}} > n_{\text{mean}},\\
H_0 &: n_{\text{MRF}} \leq n_{\text{mean}},\\
H_1 &: n_{\text{MRF}} > n_{\text{mean}},\\
Z &= \frac{T_1 - ((n_1(n_1 + n_2 + 1))/2)}{\sqrt{(n_1 n_2 (n_1 + n_2 + 1))/12}}.
\end{aligned} \tag{7}$$

As for the peak of signal-to-noise ratio (PSNR), we can use the data in Table 2 with the Wilcoxon rank-sum test and a 0.05 significance level to test the claim that the



(a)

(b)

(c)

(d)

(e)

Figure 4: Visual effects of different methods on GOPRO: (a) blurred image and results of (b) DeblurGAN, (c) DMPHN, (d) SIUN, and (e) ours. The left images are global deblur results, while local restoration details are shown on the right. Our results show clear object boundaries without artifacts and produce various generative edge maps for the discriminator D to judge the realness of the generation. The small zoom-in pictures of (e) show the good visual effect of edge attention prior and dynamic kernel selection.

Table 2: Test results of the blurred image datasets and their peak signal-to-noise ratio (PSNR) and structural similarity index (SSIM) values.

Method	GOPRO		VisDrone	
	PSNR	SSIM	PSNR	SSIM
DeepDeblur [37]	29.42370	0.761372	27.14940	0.539367
DeblurGAN [9]	28.22642	0.747912	28.29447	0.609642
DeblurGANv2 [10]	32.19638	0.87114	28.43967	0.614876
DMPHN [17]	34.21846	0.898285	28.54136	0.526301
SIUN [18]	34.46135	0.900913	28.28039	0.543417
MRFNET	*34.63429*	*0.907881*	*29.40845*	*0.862474*

multipath refinement fusion n_{MRF} has a confidence larger than mean value of other methods n_{mean}. H_0 is a hypothesis to resist the confidence zone, while H_1 is for it. Z is the specific computation to decide which is correct. The overall deviation is T_1, and n_1 and n_2 are the number of each sample. Then, the value of Z is 1.732; which is larger than 1.645 so that H_1 is in the confidence zone. In conclusion, the performance of MRFNet is better than others in terms of PSNR.

DeblurGAN required the least amount of GPU memory (equal to 4538 MB), while our proposed method required a slightly higher amount of GPU memory than DeblurGAN in GOPRO, as shown in Table 3. This is because DeblurGAN only adopts the generative network for training, which means the model is unstable and the restored color deviates from expectations, as shown in Figure 4(b). MRFNet required the least amount of GPU memory in the VisDrone dataset for a batch size of 16. The lightweight process reduced the

Table 3: Memory consumption of graphics cards.

Method	GOPRO Network (MB) + batch (8)	VisDrone Network (MB) + batch (16)
DeepDeblur	6311	7930
DeblurGAN	*4538*	6012
DeblurGANv2	6861	8107
DMPHN	6541	7329
SIUN	8399	8561
Our model	5452	*5898*

Table 4: Average time of inferring images.

Method	GOPRO Inference time(s)	Times	VisDrone Inference time(s)	Times
DeepDeblur	2.427	1.04X	2.362	1.13X
Deblur GAN	2.346	1.08X	2.144	1.24X
DeblurGANv2	2.528	1.00X	2.663	1.00X
DMPHN	1.886	1.34X	0.764	3.46X
SIUN	0.684	3.69X	0.357	7.46X
MRFNet	*0.494*	*5.12X*	*0.319*	*8.35X*

Table 5: Quantitative numerical PSNR and SSIM results.

Method	GOPRO		VisDrone	
	PSNR	SSIM	PSNR	SSIM
RefineNet [14]	34.17826	0.894369	28.73991	0.854758
LR-RefineNet	34.21445	0.906998	29.24461	0.860164
EA-RefineNet	34.39430	0.903012	29.03971	0.858601
MRFNet	*34.63429*	*0.907881*	*29.40845*	*0.862474*

number of parameters of the model and contributed to low memory usage.

MRFNet was the fastest method in terms of the time of loading the network model and inferences, as shown in Table 4. The inference was also executed on an NVIDIA RTX2080Ti GPU.

4.4. Ablation Experiments. The original MRF network used as the benchmark is denoted as RefineNet [14]. We added the lightweight and residual connection to the benchmark and denoted it as LR-RefineNet. LR-RefineNet adopts multimodel training strategy. We then added the edge reconstruction and attention modules to the refinement path on RefineNet and denoted this combination as EA-RefineNet. EA-RefineNet adopts the joint training strategy. Finally, we combined the lightweight, residual strategy, and attention modules in the benchmark and denoted combination as MRFNet. MRFNet adopts the joint training strategy as shown in Table 5; the LR-RefineNet and EA-RefineNet performed slightly better than RefineNet. MRFNet achieved the most significant numerical results.

The multiscale refinement loss function takes each subtask as an independent component within a joint task, allowing the training process to converge more rapidly and perform better than other methods. The training losses of other approaches markedly decrease during the first round and then consistently remain at a 6% smooth trend in the following training courses. The MRFNet method, aided by the loss weight scheduling technique, exhibited a dramatic downward trend initially and then remained at approximately 4%. The model accuracy improvements (approximately 10% to 21%) attributed to the multiple rounds of training for the four loss weight groups verified the convergence and advantages of our method's training strategy.

The experimental results indicate that MRFNet could achieve considerable precision. Furthermore, MRFNet executed much more quickly than other deblurring models, such as SIUN and DMPHN. Compared with DeblurGAN and DeblurGANv2, the proposed MRFNet model performed well in terms of the speed (increased by 7.4%) and deblurring quality of images (increased by 4.2%). The GPU memory use remained low owing to the added lightweight process. Our method could also recover more details and achieved relatively high SSIM and PSNR values. Images remained unstable and sometimes contained artifacts and color distortions for other models. Conversely, MRFNet was also uesd to perform image deblurring in a stable manner and resulted in high image sharpness.

4.5. Edge Attention Perception. Real-world image capture cannot avoid blurring. For instance, Figure 6(a) shows cars moving fast on a street, which causes motion blurring. The distance from the lens to the car causes a Gaussian blur. We employed the MRFNet to restore images in three steps, including edge reconstruction, localization of the blurring species, and deblurring of the patches. Edge reconstruction: edge information (high-frequency features) is very important for reconstructing images because a sharper background is beneficial for the refinement of different blurring kernels [35]. The inputs are blur and ground-truth pairs. The edge generative network then predicts the structure of the entire image. Subsequently, the pretrained networks preprocess the edge feature information to ensure that the location and class are associated with the deblurred kernels.

A broad view of edge boundaries is illustrated in Figure 6(b). The ground truth images are then preprocessed into grayscale images for further edge feature extraction and are then sent to the discriminator for the comparable benchmark. The generator produces various generative edge maps for the discriminator D to judge how real the generation is.

$$L_{\text{edge}} = \min_{G_e} \max_{D_e} L_{G_e} = \min_{G_e} \left(a_{\text{adv},1} \max_{D_e} (L_{\text{adv},1}) + a_{\text{FM}} L_{\text{AM}} \right). \tag{8}$$

Blurring category location: the attention mechanism acts in a similar manner toneural cells to focus on interesting elements using broad view [25], classification [39], and location techniques [22]. From Figures 6(e)–6(g), we can conclude that

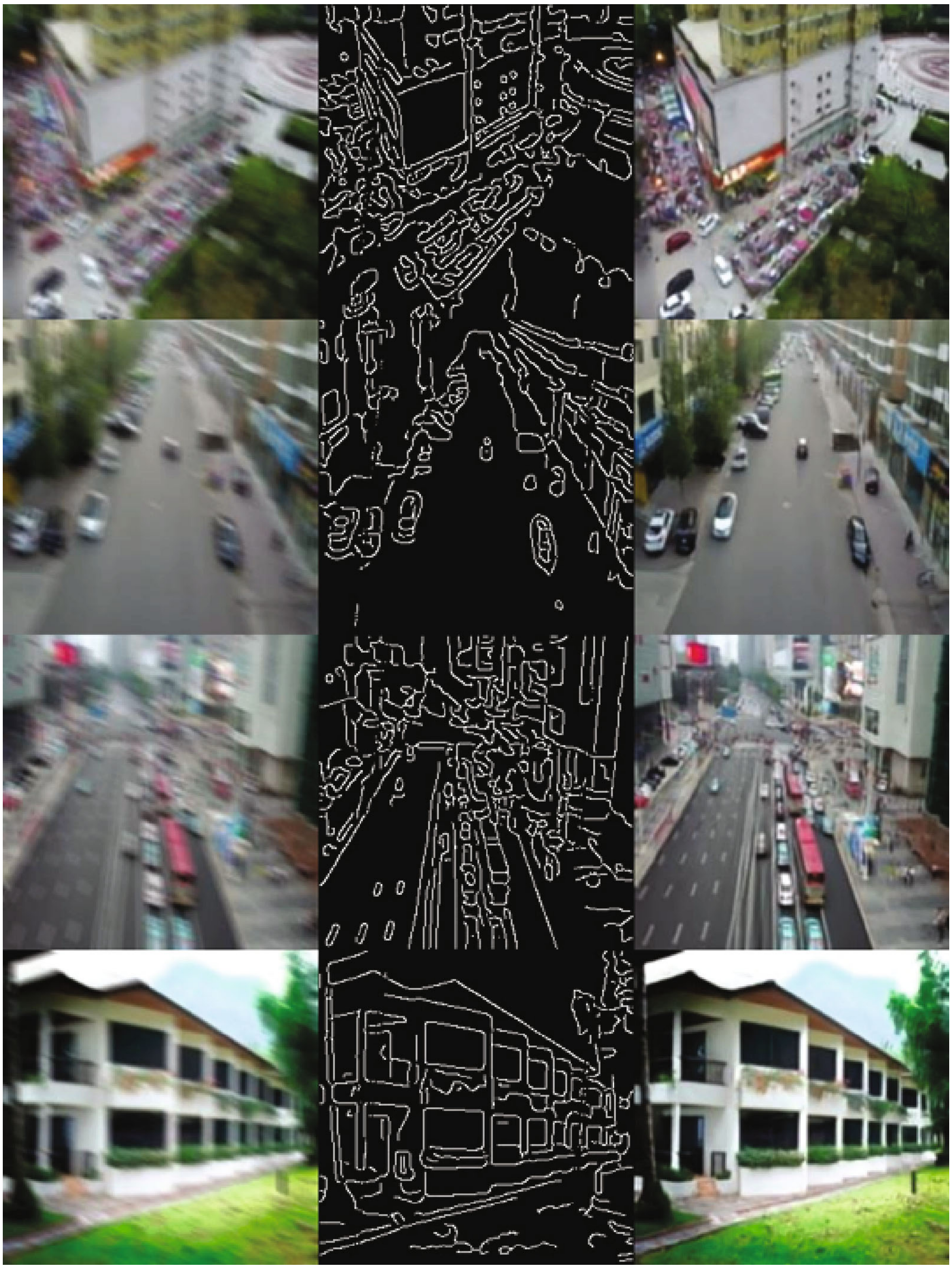


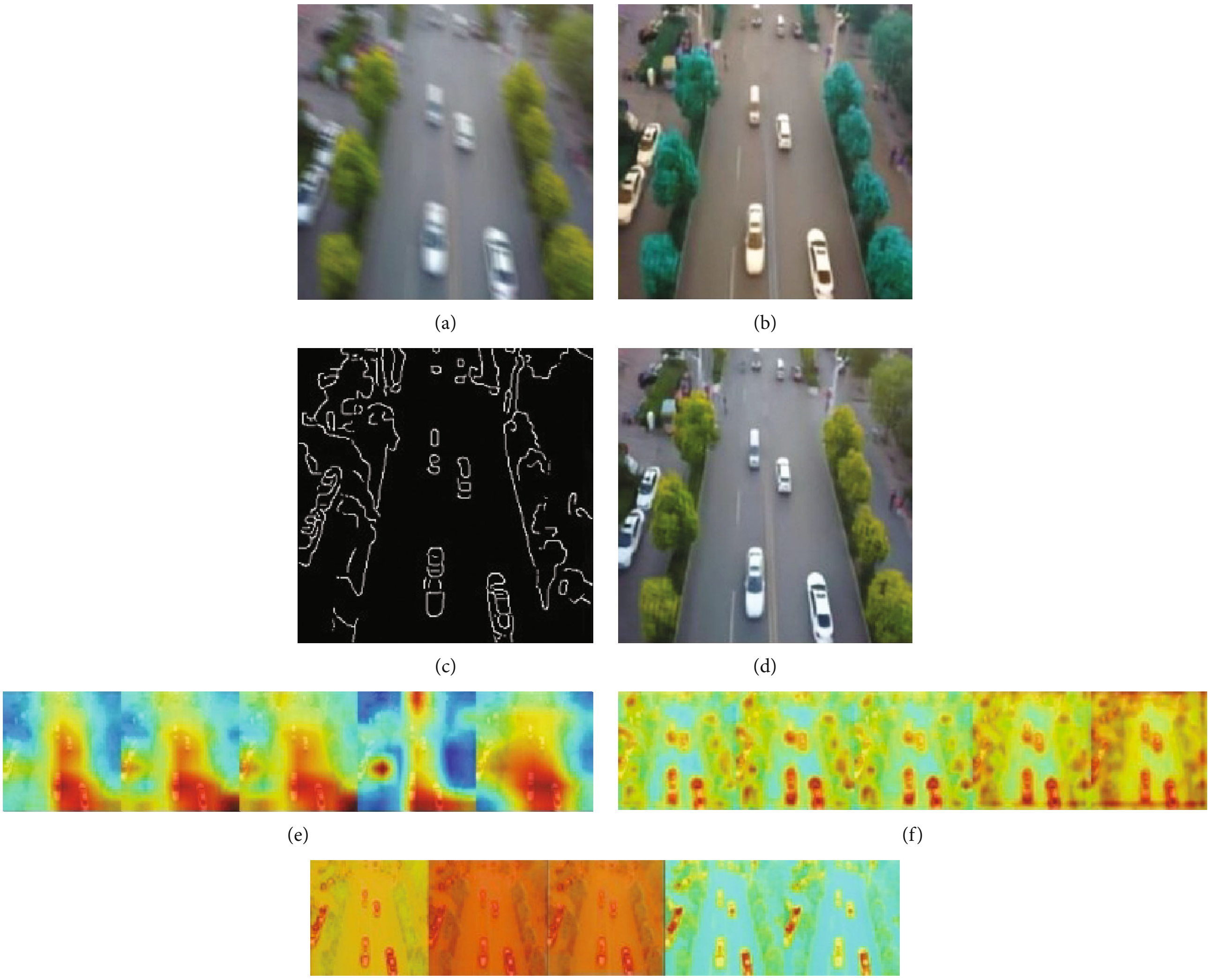
Figure 5: Edge maps and experimental results. Our restored images show vivid colors and sharp details.

changing the receptive field generates different contextual attention results. When the receptive field is large, objects are perceived in their entirety. When the receptive field is small, each object in the image is perceived and the texture is detailed.

First, we search the background using convolutional layers to create a broad view for latent meaningful objects and extract semantic information through a multipath refinement fusion unit. The second step involved classification. For a given image, $g_l(a, b)$ is the spatial information in the lth layer. G_l then represents the sum of $g_l(a, b)$. Thus, for a specific object class, the input $A_l G_l$ is the input of the softmax function. A is the weight corresponding to class, and it predicts the essential level of G_l. Finally, Q is the output of the softmax function and is denoted as $\exp(S)/\sum_e \exp(S)$.

The score S is defined as follows:

$$S = \frac{\sum A \sum g_l(a, b)}{\sum (a, b) \sum A_l \sum g_l(a, b)}. \tag{9}$$



(a) (b)

(c) (d)

(e) (f)

(g)

Figure 6: Joint generative image deblurring aided by edge attention prior is used to locate the area which is red in the picture and deblur the patches aided with edge prior. During the process of deblurring, multiple blur kernels are adopted dynamically including motion blur and Gaussian blur and so on. (a) An original blur image. (b) Attention map of specific objects. (c) Receptive field of attention is large, and the scan is with large-scale objects. (f, g) The small receptive fields, respectively. The edge recovery of a generative adversarial network (GAN) tends to be slightly intermittent, but the restoration effect is the best in complex structures. (e–g) The experimental validation of the selective regions to deblur the specific categories of blur objects in multiscale.

The score of the global average pooling predicts the importance of the location of (a, b), thus leading to the classification of a blurry object in the image.

Third, the deblurring category is located. Based on the edge maps, we can search, locate, and itemize the blurry objects into six categories, including sharp area, random deviation, changeable blur size, changeable shaking angle, changeable shaking length, and motion blurring. In terms of each category, MRFNet uses a different deblurring kernel to refine the blurring features for specific objects. The attention module was able to find and locate the general objects and apply different deblurring approaches through a deep learning training process. Subsequently, the specific objects were deblurred into sharp objects, aided by the edge generation modules and contextual attention mapping.

Patches deblurring: the structure information, predicted object, and blurry potential class could be determined when the data flow from the edge feature extraction and contextual attention were located. Subsequently, we use the deblurring feature prior network to deblur the images into sharper outputs. In this manner, we can restore the image by applying different blurring strategies in various image areas. As a result, the reconstruction of the object structure is meaningful and vivid, and the target is more specific, which improves performance.

5. Conclusions and Future Work

In conclusion, neither edge attention prior nor multimodel training can focus on the core objects in the foreground and select the proper kernels to restore. Therefore, we have designed a new algorithm consisting of three steps, including focusing, locating, and processing. The key insight of the network model and this algorithm is that the restoration of the key objects can significantly enhance the visual effects of the whole picture and retain the most semantic information. In addition, due to the selection of deblurring part of the regions with the appropriate deblur kernels rather than the whole image efficiently, the accuracy and speed are both optimized to a new level.

This study has illustrated an efficient and accurate joint edge and deblurring GAN for multifrequency feature extraction and fusion called MRFNet. This image deblurring framework uses a generative edge prior and dynamically selects proper deblurring kernels. The model is designed to overcome the challenges posed by the substantial computational resources required by CNNs and poor restoration results obtained with other methods that deal with large-scale datasets or neglect edges and color reconstruction. The proposed model has three main features for processing multiple image tasks, including color, position, and differences. Edge detectors and attention modules are then aggregated into units to refine and learn knowledge. Finally, efficient multilearning features transform a fusion into a final perceptive result.

The proposed network exploits a lightweight process, remote residual connection, edge attention mechanism, and scale refinement loss function to handle real blurring scenarios, preserving fast inference speed and high precision. It can extract different features by scheduling the weight of joint training losses and produce a fusion guided by attention modules. This leads to an efficient image restoration. The proposed MRFNet model was compared with existing models on two popular datasets for deblurring. It achieved state-of-the-art performance compared with other methods on the benchmark datasets.

In the future, we will develop a faster MRFNet model for edge computing devices. The computational capability will likely be much higher than that of the GPUs used in our experiments. The techniques of model compression, including pruning and quantization, will also be explored. This model will also be applied to video deblurring or deblurring of inpainting results at the postprocessing stage.

Data Availability

The authors declare that all data presented in this work were generated during the work and any other source has been appropriately referenced within the manuscript.

Conflicts of Interest

There are no conflicts of interests with any affiliation or person.

Acknowledgments

This research was supported by the Postgraduate Scientific Research Innovation Project of Hunan Province (Number CX20200043).

References

[1] X. Chen, Y. Zhu, W. Liu, J. Sun, and Y. Zhang, “Blur kernel estimation of noisy-blurred image via dynamic structure prior,” *Neurocomputing*, vol. 403, pp. 268–281, 2020.

[2] Q. Qi, J. Guo, and W. Jin, “EGAN: non-uniform image deblurring based on edge adversarial mechanism and partial weight sharing network,” *Signal Processing: Image Communication*, vol. 88, p. 115952, 2020.

[3] J. Rabbi, N. Ray, M. Schubert, S. Chowdhury, and D. Chao, “Small-object detection in remote sensing images with end-to-end edge-enhanced GAN and object detector network,” *Remote Sensing*, vol. 12, no. 9, p. 1432, 2020.

[4] Y. Zhang, Y. Shi, L. Ma, J. Wu, L. Wang, and H. Hong, “Blind natural image deblurring with edge preservation based on L0-regularized gradient prior,” *Optik*, vol. 225, p. 165735, 2021.

[5] Z. Fu, Y. Zheng, H. Ye, Y. Kong, J. Yang, and L. He, “Edge-aware deep image deblurring,” 2019, https://arxiv.org/abs/1907.02282.

[6] L. Xu, J. S. Ren, C. Liu, and J. Jia, “Deep convolutional neural network for image deconvolution,” *Advances in neural information processing systems*, vol. 27, pp. 1790–1798, 2014.

[7] C. J. Schuler, H. Christopher Burger, S. Harmeling, and B. Scholkopf, “A machine learning approach for non-blind image deconvolution,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1067–1074, Portland, Oregon, 2013.

[8] J. Zhang, J. Pan, W.-S. Lai, R. W. Lau, and M.-H. Yang, “Learning fully convolutional networks for iterative non-blind deconvolution,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3817–3825, Honolulu, Hawaii, USA, 2017.

[9] O. Kupyn, V. Budzan, M. Mykhailych, D. Mishkin, and J. Matas, “Deblur GAN: blind motion deblurring using conditional adversarial networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 8183–8192, Salt Lake City, Utah, U. S, 2018.

[10] O. Kupyn, T. Martyniuk, J. Wu, and Z. Wang, “DeblurGAN-v2: deblurring (orders-of-magnitude) faster and better,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 8877–8886, Seoul, Korea, 2019.

[11] K. Zhang, W. Zuo, S. Gu, and L. Zhang, “Learning deep CNN denoiser prior for image restoration,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3929–3938, Honolulu, Hawaii, USA, 2017.

[12] S. Zheng, Z. Zhu, J. Cheng, Y. Guo, and Y. Zhao, “Edge heuristic GAN for non-uniform blind deblurring,” *IEEE Signal Processing Letters*, vol. 26, no. 10, pp. 1546–1550, 2019.

[13] S. Nah, T. Hyun Kim, and K. Mu Lee, “Deep multi-scale convolutional neural network for dynamic scene deblurring,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3883–3891, Honolulu, Hawaii, USA, 2017.

[14] G. Lin, A. Milan, C. Shen, and I. Reid, “RefineNet: multi-path refinement networks for high-resolution semantic segmentation,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5168–5177, Honolulu, Hawaii, USA, 2017.

[15] X. Tao, H. Gao, X. Shen, J. Wang, and J. Jia, “Scale-recurrent network for deep image deblurring,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8174–8182, SALT LAKE CITY, 2018.

[16] V. Nekrasov, C. Shen, and I. Reid, “Light-weight RefineNet for real-time semantic segmentation,” 2018, https://arxiv.org/abs/1810.03272.

[17] H. Zhang, Y. Dai, H. Li, and P. Koniusz, “Deep stacked hierarchical multi-patch network for image deblurring,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5978–5986, Long Beach, California, USA, 2019.

[18] M. Ye, D. Lyu, and G. Chen, “Scale-iterative upscaling network for image deblurring,” *IEEE Access*, vol. 8, pp. 18316–18325, 2020.

[19] Q. Shan, J. Jia, and A. Agarwala, “High-quality motion deblurring from a single image,” *ACM Transactions on Graphics*, vol. 27, no. 3, pp. 1–10, 2008.

[20] D. Krishnan, T. Tay, and R. Fergus, “Blind deconvolution using a normalized sparsity measure,” in *CVPR 2011*, pp. 233–240, Colorado Springs, CO, USA, 2011.

[21] L. Xu and J. Jia, “Two-phase kernel estimation for robust motion deblurring,” in *European conference on computer vision*, pp. 157–170, Berlin, Heidelberg, 2010.

[22] Z. Hu and M. H. Yang, “Learning good regions to deblur images,” *International Journal of Computer Vision*, vol. 115, no. 3, pp. 345–362, 2015.

[23] Y. Fang and T. Zeng, “Learning deep edge prior for image denoising,” *Computer Vision and Image Understanding*, vol. 200, p. 103044, 2020.

[24] J. Pan, Z. Hu, Z. Su, and M.-H. Yang, “Deblurring text images via L0-regularized intensity and gradient prior,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2901–2908, Columbus, Ohio, 2014.

[25] M. D. Zeiler and R. Fergus, “Visualizing and understanding convolutional networks,” in *European conference on computer vision*, pp. 818–833, Zürich, 2014.

[26] Q. Feng, H. Fei, and W. Wencheng, “Blind image deblurring with reinforced use of edges,” *The Visual Computer*, vol. 35, no. 6-8, pp. 1081–1090, 2019.

[27] T. A. Javaran, H. Hassanpour, and V. Abolghasemi, “Non-blind image deconvolution using a regularization based on re-blurring process,” *Computer Vision and Image Understanding*, vol. 154, pp. 16–34, 2017.

[28] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, “Object detectors emerge in deep scene CNNs,” in *International Conference on Learning Representations*, The Hilton San Diego Resort & Spa, 2015.

[29] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, “Is object localization for free? Weakly-supervised learning with convolutional neural networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, Boston, Massachusetts, 2015.

[30] O. Whyte, J. Sivic, A. Zisserman, and J. Ponce, “Non-uniform deblurring for shaken images,” *International Journal of Computer Vision*, vol. 98, no. 2, pp. 168–186, 2012.

[31] M. Hirsch, C. J. Schuler, S. Harmeling, and B. Schölkopf, “Fast removal of non-uniform camera shake,” in *2011 International Conference on Computer Vision*, pp. 463–470, Barcelona, Spain, 2011.

[32] O. Whyte, J. Sivic, and A. Zisserman, “Deblurring shaken and partially saturated images,” *International Journal of Computer Vision*, vol. 110, no. 2, pp. 185–201, 2014.

[33] J. Dai and Y. Wang, “Multi-scale residual convolution neural network and sector descriptor-based road detection method,” *IEEE Access*, vol. 7, pp. 173377–173392, 2019.

[34] K. Schelten, S. Nowozin, J. Jancsary, C. Rother, and S. Roth, “Interleaved regression tree field cascades for blind image deconvolution,” in *2015 IEEE Winter Conference on Applications of Computer Vision*, pp. 494–501, Waikoloa Beach, Hawaii, USA, 2015.

[35] 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*, Nevada, United States, 2016.

[36] Q. Wang, S. Shi, S. Zheng, K. Zhao, and X. Chu, “FADNet: a fast and accurate network for disparity estimation,” in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 101–107, Paris, France, 2020.

[37] J. Mei, Z. Wu, X. Chen, Y. Qiao, H. Ding, and X. Jiang, “DeepDeblur: text image recovery from blur to sharp,” *Multimedia Tools and Applications*, vol. 78, no. 13, pp. 18869–18885, 2019.

[38] P. Zhu, D. Du, L. Wen et al., “Vis Drone-VID 2019: the vision meets drone object detection in video challenge results,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pp. 227–235, Seoul, Korea, 2019.

[39] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, “Learning deep features for discriminative localization,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, Nevada, United States, 2016.

Hindawi
Wireless Communications and Mobile Computing
Volume 2021, Article ID 9974270, 35 pages
https://doi.org/10.1155/2021/9974270

Research Article

Design and Development of an Efficient Network Intrusion Detection System Using Machine Learning Techniques

Thomas Rincy N[1] **and Roopam Gupta**[2]

[1]*Department of Computer Science and Engineering, University Institute of Technology, Rajiv Gandhi Proudyogiki Vishwavidyalaya, Bhopal, M.P, India*
[2]*Department of Information Technology, University Institute of Technology, Rajiv Gandhi Proudyogiki Vishwavidyalaya, Bhopal, M.P, India*

Correspondence should be addressed to Thomas Rincy N; rinc_thomas@rediffmail.com

Received 17 March 2021; Revised 16 April 2021; Accepted 14 May 2021; Published 28 June 2021

Academic Editor: Philippe Fournier-Viger

Today's internets are made up of nearly half a million different networks. In any network connection, identifying the attacks by their types is a difficult task as different attacks may have various connections, and their number may vary from a few to hundreds of network connections. To solve this problem, a novel hybrid network IDS called NID-Shield is proposed in the manuscript that classifies the dataset according to different attack types. Furthermore, the attack names found in attack types are classified individually helping considerably in predicting the vulnerability of individual attacks in various networks. The hybrid NID-Shield NIDS applies the efficient feature subset selection technique called CAPPER and distinct machine learning methods. The UNSW-NB15 and NSL-KDD datasets are utilized for the evaluation of metrics. Machine learning algorithms are applied for training the reduced accurate and highly merit feature subsets obtained from CAPPER and then assessed by the cross-validation method for the reduced attributes. Various performance metrics show that the hybrid NID-Shield NIDS applied with the CAPPER approach achieves a good accuracy rate and low FPR on the UNSW-NB15 and NSL-KDD datasets and shows good performance results when analyzed with various approaches found in existing literature studies.

1. Introduction

Research in network security is a vastly emerging topic in the domain of computer networking due to the ever-increasing density of advanced cyberattacks. The intrusion detection systems (IDSs) are designed to avert the intrusions and to protect the programs, data, and illegitimate access of the computer systems. The IDSs can classify the intrinsic and extrinsic intrusions in the computer networks of an organization and instigate the alarm if security infringement is comprised in an organization network [1]. One of the notable definitions for intrusion is that it produces malignant, outwardly activated functional violations. The primary goal of intrusion detection systems is to recognize a broad variety of intrusions, heretofore identified and unidentified attacks; to discover and adapt to unfamiliar attacks; and to detect and recognize intrusions in a prompt pattern [2]. The preliminary work on IDSs was researched by Anderson [3] who recommended means of examining data. Subsequent to Anderson's work, the previous work was aimed at developing the algorithms and procedures for online automated systems. The Sytek project [4] started producing audit trails having enhanced security and considered different approaches for analyzing automated systems. These observations contributed to the first empirical evidence that the end users can be recognized from each other through user action of using the computer [5]. The proof of SRI and Sytek studies [6] was the foundation of real-time IDS. The behavior of the users, whether it is normal or suspected, is continuously monitored by these systems. The real-time IDS relies on two techniques: (1) intrusions whether normal or suspected can be tracked by the flagged departure from the factual patterns of respective users and (2) perceived system susceptibilities and various infractions of the system-aimed security protocols are best tracked from rule-based expert systems. The stability of precision and detection is primarily two measures applied mainly to assess the IDSs [7], and in recent years, many IDS research surveys have been accomplished to enhance these measures [8]. In the inception stages, many of the research studies mainly focus on the rule-based expert system and statistical approach. However, the various performance results show that these approaches when applied to large datasets are not accurate and precise [9].

To get the better of the above-mentioned problem, data mining approaches [10, 11] and machine learning techniques were introduced [12]. Some machine learning paradigms containing Graph-based methods [13], Linear Genetic Programming [14], Bayesian Network [15], k-NN [16], K-means clustering [17], Hidden Markov Model [18], Self-organizing map [19], etc. have been explored for the architecture of IDSs. Machine learning [20] can detect the correlation between features and classes found in training data and identify relevant subsets of attributes by feature selection and dimensionality reduction, then use the data to build a model for classifying data to perform predictions. The data dimensionality related to data mining and machine learning has doubled in the last decade that leads to several questions to current learning approaches [21]. Due to the presence of excessive cardinal features, the model that tends to learn gets overfitted, resulting in the performance degradation of the model.

To solve the problem of data dimensionality in machine learning and data mining, various dimensionality reduction approaches have been accessed which is considered as an essential step in the area of machine learning and data mining. Feature selection is an extensively employed and efficient technique applied for dimensionality reduction. The main aim of feature selection is to select the limited feature subsets from primary features conferring to relevancy appraisal standard that manages the training model to accomplish greater performance outcomes and reduced execution time and achieve higher model predictability. Most of the classification problem needs the supervised learning where the class-conditional possibilities and cardinal class are not familiar and the class labels and its instances are associated with each other [22]. There is a scarcity of knowledge in real-world applications related to relevant features. Endless feature candidates are acquired to generate the more coherent domain, which results in the existence of irrelevant and redundant features to the target approach or objective function. For the target approach, the relevant or significant features are not irrelevant or redundant; neither the redundant feature is spontaneously correlated with the target approach or objective function but impacts the learning approach. The new events are not added by the redundant features to the target approach or objective function. In the majority of the classification problems, it is a composite to learn even if the classifier is competent due to the presence of an enormous number of data, till the redundant features are excluded from the objective function. For the classification problem, the features are once generated then instead of processing with full data; the feature selection will bring about the feature subsets from the initial features and then process with the feature subsets to a learning algorithm. The nominally sized feature subsets for the classification problem are selected by the feature selection approach conferring to the following criterion:

(i) Normally, the classifier accuracy does not decline considerably

(ii) Among all the likely features, the initial distribution of the class shall be approximately close to the proceeding distribution of the class whenever the values are likely towards the features selected

To obtain the high merit feature subsets from 2^m subsets, the feature subset selection approaches search feature subsets conferring to a few significant appraisal criteria. However, this approach is intensive for the conclusion of the best subset and to select the intermediate-sized feature subsets with the volume (m); the strategy is expensive and restrictive. Various approaches like heuristic and random search lower the computational intricacy by a trade-off. To prohibit the feature subsets from exhaustively searching, a stopping criterion is required. A feature selection approach [23] does the job by subset generation, subset evaluation, stopping criterion, and the result from the validation. With the likely search approach, the chosen feature subsets are sent for subset evaluators with significant evaluation criteria. After the stopping criterion is performed, the feature subset that is competent enough to fit in the evaluation strategy is preferred, and then, finally, the finest feature subset is selected and gets authenticated by employing the domain knowledge or validation.

The detection methods of intrusion detection systems are classified into three major types: anomaly-based, signature-based, and hybrid-based. The signature-based IDS and anomaly-based IDS were the most favored methods in an organization until numerous shortcomings were observed, which leads to the development of hybrid intrusion detection systems. In the designing of IDS, classifying the datasets according to attack types and selecting the good feature subsets are a hard problem. The classifying of datasets according to attack types aids in predicting the vulnerability of individual attacks in various networks. Moreover, relevant features should not be irrelevant or redundant so that accurate and highly merit feature subsets are obtained. To address this issue, a new hybrid network intrusion detection system called NID-Shield is designed that classifies the dataset according to attack types. Furthermore, the hybrid CAPPER approach is applied as a feature subset selection approach. Screening is applied to those features by the CAPPER approach which is redundant having a high-class correlation. Moreover, machine learning algorithms are applied for selecting high merit and accurate feature subsets.

The major contributions of this manuscript are as follows:

(i) An efficient hybrid NID-Shield NIDS is proposed in this manuscript that classifies the UNSW-NB15 and NSL-KDD datasets according to the attack types and attack names

(ii) An effective hybrid feature subset selection method called CAPPER is applied as a feature subset selection that combines the CFS and Wrapper approaches for obtaining the reduced accurate and high merit feature subsets

(iii) The reduced accurate and high merit datasets obtained from CAPPER are trained by the machine learning approaches and assessed by a 10-fold cross-validation method

(iv) The hybrid NID-Shield network intrusion detection system shows overall good improvement results on the different approaches found in the existing literature studies

The remaining article is coordinated accordingly. Section 2 focuses on related work. Section 3 proposes the architecture of the hybrid NID-Shield NIDS. Section 4 relates to the characteristics of UNSW-NB15 and NSL-KDD datasets. Section 5 discusses the performance evaluation of the hybrid NID-Shield NIDS approach with various existing approaches on the UNSW-NB15 and NSL-KDD dataset, and Section 6 then concludes the work.

2. Related Work

This section introduces the existing literature studies on the hybrid network intrusion detection system. Moreover, this section discusses the advantage of a hybrid intrusion detection system over a traditional intrusion detection system. Furthermore, distinct machine learning approaches are acquainted and discuss the usefulness of selecting specific machine learning techniques.

2.1. State-of-the-Art Network IDSs. The research in the manuscript is focused on studying the appropriateness of intrusion detection approaches to recognize network-level intrusions, as the network structures generate resources more susceptible to intrusions than autonomous machines. Three facets of network structures generate resources more exposed to attack by an autonomous machine: (1) networks generally provide additional resources than autonomous machines; (2) networks are usually formed to aid resource sharing; and (3) the global security policies that are applied to the IDS are limited [24]. Moreover, the hybrid methods are suggested over the signature and anomaly-based IDS, as the integration of multiple approaches into a distinct hybrid system retains the advantages of multiple techniques, while reducing many of the deficiencies [25].

Acharya and Singh [26] conclude that for obtaining the best possible detection and accuracy rate, the hybrid learning approaches can be a good choice and proposed intelligent water drop (IWD) algorithm, introduced by Shah–Hosseini [27]. This approach applies the support vector machine (SVM) as a classification algorithm and IWD approach as a feature selection technique that is inspired by the nature. IWD approach selects the best feature subsets, and the evaluation of the subsets is executed by the SVM classifier. The proposed model lowers the forty-five features from the applied dataset to the lowest of ten features. KDD-Cup ‘99 dataset is used for the appraisal of metrics. The proposed approach attains an accuracy, detection, and precision of 99%. The disadvantage of applying the elemental IWD algorithm is the likelihood of choosing the adjacent node for a water drop to stream.

Arif et al. [28] introduced the hybrid approach for IDSs. In this approach, pruning of the node is performed by PSO and pruned decision tree is applied for the classification purpose in a NIDS. The proposed approach applies the single and multiple-objective particle swarm optimization (PSO) algorithms. The KDD-Cup ‘99 dataset is used as an experimental evaluation approach. From the 10% KDD-Cup ‘99 training and testing dataset, thirty arbitrary samples are chosen for evaluation purposes. The statistical records in every training and testing dataset are 12,000 and 24,000 accordingly for the appraisal of the metrics. The precision of 99.95% and accuracy of 93.5% are achieved using the above approaches. But there are some primary problems involved with traditional PSO when adopted as a feature selection approach. The most significant problem submits the following question: in a random initialization, from the initial population, how far is it to reach an optimal solution. If the optimum answer tells that the predicted prediction is far distant, then it may not be possible to obtain the global optimal solution within the allocated time. The second problem involves the conventional upgrading mechanism of global best and personal best of the PSO approach, as these mechanisms may result in losing some valuable features.

Ahmed et al. [29] applied a triple strategy to build a hybrid IDS in which the Naive Bayes feature subset selector (NBFS) technique has been applied for dimensionality reduction. For the outlier rejection, optimized support vector machines (OSVM) are applied, whereas prioritized k-nearest neighbors (PKNN) are applied as a classifier. The NSL-KDD, KDD-Cup ‘99, and Kyoto 2006+ datasets are used for evaluation purposes. 18 efficient features are preferred from the KDD-Cup ‘99 dataset with a detection ratio of 90.28%. 24 features are selected from the Kyoto 2006+ dataset having a detection ratio of 91.60%. The author has compared with previous work and has the best overall detection ratio of 93.28%. The major disadvantage with the Naive Bayes is that it presumes prediction of the features that are mutually independent to one another. The features with mutual independence are consistently hard to get in real-world problems.

Dash et al. [30] reports two new hybrid intrusion detection methods that are GS and sequence of GSPSO which is the combination of gravitational search and the particle swarm optimization algorithms. It involves search agents who relate to each other having heavy masses from the gravitational force, and their performance is assessed by their mass. The combination approach has been carried out to train ANN with models such as GS-ANN and GSPSO-ANN. The random selection of 10% features is selected for training purposes, while 15% is used for testing purposes and is applied successfully for intrusion detection purposes. The author does not apply any feature selection technique. The KDD-Cup ‘99 dataset was applied as a metric for

Table 1: Taxonomy of latest hybrid intrusion detection methods.

Hybrid-based intrusion detection techniques with feature selection techniques							
Year	Research papers	Algorithms	Techniques	Dataset	Evaluation criteria	Feature selection	Results
2017	[26]	SVM, IWD	SVM is applied as a classifier. Feature reduction applying IWD (intelligent water drop) method	KDD-Cup ‘99 dataset	Detection rate, precision rate, accuracy rate, false alarm rate	IWD	Achieves a detection rate of 99.40%, precision rate of 99.10%, false alarm of 1.40%, accuracy rate of 99.05%
2017	[28]	Particle swarm optimization (PSO)	Particle swarm optimization (PSO) algorithm is applied for pruning the node of DT, and the pruned DT is applied for the network IDS classification	KDD-Cup ‘99 dataset	Accuracy rate, precision rate, FPR., IDR, time	PSO	Accuracy of 96.65%, a precision of 99.98%, FPR of 0.136, IDR of 92.71%, and execution time of 383.58 sec. is obtained
2017	[29]	Prioritized KNN algorithm, optimized SVM algorithm, Naïve Bayes feature selection approach	PKNN is used for detecting input attacks, hybrid HIDS strategy (based on Naïve Bayes feature selection); OSVM is applied for outlier rejection. Naïve Bayes is applied as the feature selector approach	Kyoto 2006+ dataset, KDD-Cup ‘99 dataset, and NSL-KDD dataset	Specificity, sensitivity, detection rate, precision	NBFS	An overall sensitivity rate of 53.24%, detection rate of 94.6%, precision of 56.62%, specificity of 98.21% are obtained on all datasets
2017	[30]	Artificial neural network	Particle swarm optimization (GSPSO) is employed to train ANN, gravitational search (GS), and combination of GS	NSL-KDD dataset	MSE, detection rate, time	Not applied	MSE of 0.4527%, a detection ratio of 95.26%, and execution time of 103.70 seconds are obtained
2017	[31]	Hybrid multilevel data mining algorithm	Flexible mutual information-based feature selection (FIMS) is employed as feature selector, MH-ML (multilevel hybrid machine learning), MH-DE (multilevel hybrid data engineering), MEM (micro expert module) for training the KDD-Cup ‘99 dataset	KDD-Cup ‘99 dataset	Detection rate, recall, accuracy rate, *F*-value, precision rate	FIMS	A detection rate of 66.69%, accuracy of 96.70%, recall of 96.70%, precision of 96.55%, and *F*-value of 96.60% are achieved
2018	[32]	Support vector machine (SVM)	Chisqselector employing the SVM classifier for reduction of features	KDD-Cup ‘99 dataset	AUPR, AUROC, time	Chisqselector	AUPR of 96.24%, AUROC of 99.55%, and execution time of 10.79 seconds are obtained

Table 1: Continued.

Hybrid-based intrusion detection techniques with feature selection techniques							
Year	Research papers	Algorithms	Techniques	Dataset	Evaluation criteria	Feature selection	Results
2018	[33]	Vector-based genetic algorithm	Three feature selection methods are employed, linear correlation-based feature selection (LCFS), modified mutual information-based feature selection (MMIFS), and forward feature selection algorithm (FFSA), chromosomes as vector and training data as metrics	KDD-Cup '99 dataset and CTU-13 dataset	FPR, accuracy rate	LCFS, FFSA, MMIFS	FPR of 0.17% is achieved, and accuracy rate for the DoS is 99.8%
2018	[34]	Neural network with resilient back propagation algorithm, CART	Neural network with resilient back propagation algorithm to update the weights; feature reduction is performed by CART	ISCX & ISOT dataset	Detection rate, accuracy rate, FPR	CART	An accuracy rate of 99.20%, detection rate of 99.08%, and FPR of 0.75% are obtained
2018	[35]	Symmetrical uncertainty and genetic algorithm (SU-GA) is used as classification algorithm	Genetic algorithm is used on selected features; symmetric uncertainty is applied to find best features	UCI dataset	Accuracy rate	GA	An accuracy of 83.83% is obtained, and an execution time of 0.23 seconds is achieved on all approaches
2018	[36]	Genetic algorithm	Neurofuzzy inference system, neural fuzzy genetic, fuzzy logic controller, multilayer perception for attack classification	KDD-Cup '99 dataset	Accuracy rate	Fuzzy rule	A true attack detection and false alarm detection accuracy up to 99% rate of 1%.
2019	[37]	Random forest, Naive Bayes, J-48, k-nearest neighbor algorithm	WrapperSubsetEval and CfsSubsetEval are applied as two feature selection techniques, while random forest, k -NN algorithm, Naive Bayes, and J-48 are applied as the classifiers	NSL-KDD dataset	Detection rate, accuracy rate, F– measure, TP rate, FP rate, MCC, and time	Wrapper and filter	Overall accuracy rate of 99.86%, overall FPR of 0.00035%, overall detection ratio of 0.9828%, F-measure of 0.706%, overall TPR of 0.929%, overall MCC of 0.955%, and total execution time of 10.625 seconds (executed on NSL-KDD dataset with 25 attributes on all attack types)
2019	[38]	K-means clustering, DBSCAN, SMO	K-means is applied for data grouping, DBSCAN is employed to eliminate noise from data, and SMO is applied for intrusion detection	KDD-Cup '99 dataset	Detection rate, accuracy rate	DBSCAN	An approx detection rate of 70% and an approx accuracy of 98.1% are obtained

Table 1: Continued.

Hybrid-based intrusion detection techniques with feature selection techniques							
Year	Research papers	Algorithms	Techniques	Dataset	Evaluation criteria	Feature selection	Results
2019	[39]	Intelligent flawless feature selection algorithm (IFLFSA), entropy-based weighted outlier rejection (EWOD), intelligent layered classification algorithm	EWOD is used to detect outliers in data, IFLFSA is used as feature selection, and intelligent layered classification algorithm is applied to classify the data	KDD-Cup '99 dataset	Accuracy rate	IFLFSA	Overall accuracy of 99.45% is achieved
2019	[40]	ID3, *k*-nearest neighbor, isolation forest	*k*-nearest neighbor is used to apply a class to unknown data point, ID3 is used as feature selector, and isolation forest is employed to segregate normal data from anomaly	NSL-KDD & KDD-Cup '99 dataset	Detection rate, accuracy rate, false alarm rate	*k*-NN	The performance with KDD-Cup '99 dataset has a detection rate of 97.20%, accuracy of 96.92%, and FPR of 7.49%. Performance on NSL-KDD dataset has a detection rate of 95.5%, accuracy of 93.95%, and a FPR of 10.34%
2019	[41]	Best first search and Naïve Bayes (BFS-NB) algorithm	Best search is applied as attribute optimization approach, and Naïve Bayes is employed as classifier	KDD datasets from the US Air Force	Accuracy, sensitivity, specificity	Naive Bayes	Sensitivity analysis of 97%, accuracy of 92.12%, and specificity of 97.5% are obtained
2020	[42]	Deep neural network (DNN), classical AutoEncoder (CAE)	Deep neural network (DNN) is applied as classification, and classical AutoEncoder (CAE) is applied as a feature selector approach	UNSW-NB15 dataset	(DNN)	Classical AutoEncoder (CAE)	Precision of 92.08%, *F* -measure of 91.35%, accuracy of 91.29%, recall of 90.64%, and FPR of 0.805
2020	[43]	*k*-nearest neighbor (*k* -NN), extreme learning machine (ELM), hierarchical extreme learning machine (H-ELM), SDN controller	Hierarchical extreme learning machine (H-ELM), extreme learning machine (ELM), and *k*-nearest neighbor (*k*-NN) are applied for classification, and SDN controller is employed as a feature selection approach	NSL-KDD dataset	(*k*-NN), (ELM), (H-ELM)	SDN controller	An accuracy of 84.29%, FPR of 6.3%, precision of 94.18%, recall of 77.18%, *F* -measure of 84.83%
2021	[44]	ANN is applied as a classifier	An integration technique (CFS + ANN) is employed to improve the classification accuracy	NSL-KDD dataset and UNSW-NB15 dataset	(CFS + ANN)	Correlation-based feature selection technique	An accuracy of 98.45%, specificity of 94.38%, sensitivity of 92.94%, and execution time of 500 seconds are obtained on the NSL-KDD dataset. For the UNSW-NB15 dataset, an accuracy of 96.44%, specificity of 98.4%, a sensitivity of 50.4%,

Table 1: Continued.

Hybrid-based intrusion detection techniques with feature selection techniques							
Year	Research papers	Algorithms	Techniques	Dataset	Evaluation criteria	Feature selection	Results
							and an execution time of 1023 seconds are achieved
2021	[45]	SVM, modified binary gray wolf algorithm	SVM is used as a classifier and, modified binary gray wolf algorithm is applied as feature selection approach	NSL-KDD dataset	SVM	Modified binary gray wolf algorithm	An accuracy of 96%, FPR of 0.03, detection rate of 0.96, and execution time of 69.6 h
2021	[46]	Multiclassifier, deep neural network, kernel density	Random forest differential evaluation with kernel density for predicting unusual activities. For input classification, a multiclassifier is applied, while a deep neural network is employed as the learning and training of the data. Kernel density is used for clustering and prediction of data.	HHAR dataset	Random forest differential evaluation with kernel density, multiclassifier, deep neural network, kernel density	Basic sort-merge tree	An accuracy rate of 98.4%, a sensitivity of 96.02%, and a specificity of 99.8%

calculation. Normalization of the dataset was done for uniform distribution by MATLAB. An average detection ratio of 95.26% was achieved. The gradual shift of the search agent encourages the relevant solution of the algorithm, but the major weakness is its speed of convergence that slows down in subsequent stages and has the tendency to get trapped in the local optimum solution.

Yao et al. [31] introduced a hybrid framework for IDS. K -means algorithm is employed for clustering purposes. In the classification phase, many machine learning algorithms (SVM, ANN, DT, and RF) which are all supervised learning algorithms are compared on different parameters. The supervised learning algorithm has various parameters for different kinds of attacks (DoS, U2R, Probe, and R2L). FIMS is applied as a feature selection technique. The proposed approach has obtained an accuracy rate reaching 96.70% with the KDD-Cup ‘99 dataset. The drawback with the FIMS approach is that the correlation between the candidate features and their class is not considered.

Suad and Fadl [32] introduced an IDS model applying the machine learning algorithm to the big data environment. This paper employs a Spark-Chi-SVM model. ChisqSelector is applied as a feature selection method, and an IDS model is constructed by applying the SVM as a classifier. The comparison is done with the Spark-Chi-SVM classifier and Chi-logistic-regression classifier. The KDD-Cup ‘99 dataset is used for the metrics of the evaluation process. The result shows that the Spark-Chi-SVM model shows good performance having an AUROC of 99.55% and an AUPR of 96.24%. The disadvantage of ChisqSelector is having a larger sensitiveness towards the sample size. However, when the sample size increases, the total differences become smaller than the predicted value.

Ijaz et al. [33] introduce a genetic algorithm, which is based on vectors. In this technique, vector chromosomes are applied. The uniqueness of this algorithm is that it shows the chromosomes as a vector and training data as metrics. It grants multiple pathways to have a fitness function. Three feature selection techniques are chosen: forward feature selector algorithm (FFSA), linear correlation feature selector (LCFS), and modified mutual information feature selector (MMIFS). The novel algorithm is tested in two datasets (CDU-13 and KDD-Cup ‘99). Performance metrics demonstrate that the vector genetic algorithm has a high detection ratio of 99.8% and a low false positive rate of 0.17% on the denial of service (DoS) attack. However, the authors do not evaluate the U2R, Probe, and R2L attacks which are considered important metrics in the IDS.

Alauthaman et al. [34] proposed an approach of peer-to-peer bot detection build on a feed-forward neural network in assistance with the DT. CART is then applied as a feature selection approach to obtain the significant features. Network traffic reduction techniques were applied by using six rules to pick the most relevant features. Twenty-nine features are selected from six rules. The proposed approach obtained an accuracy of 99.20% and a detection ratio of 99.08%, respectively. The disadvantage of utilizing a CART is that the decision tree may not be stable and the CART splits the variables one by one.

Venkataraman and Selvaraj [35] report an efficient hybrid feature selection structure for the classification of the data. For classification purposes, symmetrical uncertainty is applied to find the relevant features. Moreover, GA is applied to search for the merit subset with higher accuracy. The author combined SU-GA as a hybrid feature selection approach. MATLAB and Weka tools are applied for evaluation purposes. Different classification algorithms (KStar, J48, NB, SMO, DT, JRIP, Multilayer Perceptron, and Random forest) are used to classify different attacks. The average learning accuracy with Multi Perpn and SU-GA is the highest having 86.0%. The major drawback of a genetic algorithm is that it may be computationally expensive, as the training of the model is required for the appraisal of each candidate. GA is stochastic, so it may require a longer time to converge.

Kumar and Kumar [36] introduce an intelligent-based hybrid NIDS model. This model then integrates the multilayer perception, fuzzy logic controller, adaptive neurofuzzy interference system, and a neurofuzzy genetic. The author applied fuzzy logic as a feature selection method. The proposed system has three key elements: analyzer, collector, and predictor modules, for gathering and filtering network traffic to classify the data and prepare the final decision in assuming knowledge on the accurate attack. The experiment is assessed on the KDD-Cup '99 dataset that achieves an improvement of true attack detection and false alarm detection accuracy upto 99% rate of 1% using MATLAB. The disadvantage of fuzzy logic is that the results are observed based on assumptions, and due to this reason, accuracy is sometimes incorrect.

Cavusoglu et al.[37] applied the hybrid approach for IDS using machine learning techniques. *k*-nearest neighbor and Naive Bayes algorithms are used for classification purposes, while the random forest algorithm is used as a classifier. The author applied two feature selection techniques called the CfsSubsetEval and WrapperSubsetEval approach. J48 algorithm is applied in conjunction with WrapperSubsetEval for selecting accurate attributes. For the evaluation of metrics, the NSL-KDD dataset is applied. The overall accuracy of 99.86% is obtained on all types of attacks.

Saxena et al. [38] implemented a DBSCAN-based hybrid technique for obtaining the high-quality feature subsets for IDS. DBSCAN is employed as a method for eliminating noise from data. For grouping data, *K*-means clustering is proposed. The SMO classifier is applied for classification purposes. The KDD-Cup '99 dataset is applied for evaluation purposes with reduced attributes. The proposed approach, DBKSMO, achieved an accuracy of about 98%. Weka and MATLAB tools are applied for the execution of the results. However, the major disadvantage of DBSCAN is that whenever there is a cluster having variations in density or the clusters having similar variation, its performance declines, the major reason being the setting of ε (distance threshold), and minimum points for determining the neighborhood points will change from clusters to clusters, whenever density changes. This problem exists for high-dimensional data, as the ε (distance threshold) becomes difficult to examine.

Kambattan and Rajkumar [39] introduced effective IDS, which employs a feature selection technique named IFLFSA to select the finest reduced features that are effective for analyzing the attacks. To identify the outliers from the dataset, the EWOD approach is utilized. An intelligent layered technique is employed for efficient classification. For experimental purposes, the KDD-Cup '99 dataset is applied. The comprehensive detection rate wraps the detection rate on four types of attacks, namely, Probe, DoS, U2R, and R2L. The detection rate of the proposed system is achieved at a rate of 99.45%. The weakness of using intelligent agents is that whenever the global constraints are applied, the intelligent agent fails to deliver appropriately. Each agent is more effective in dealing individually with the main or central controller. The agents make the decisions based on locally acquired knowledge; whenever there is global knowledge available, the agents are missing the major available knowledge globally.

Kar et al. [40] utilize the decision tree algorithm called ID3 which is applied for the classification of the data into its corresponding classes. To designate the class labels to its unexplored data point on its class labels to the *k*-nearest point, the *k*-NN approach is applied. Isolation forest is introduced to isolate the anomaly against normal instances. The suggested approach HFA has applied to the NSL-KDD and KDD-Cup '99 dataset. The metrics on the KDD-Cup '99 dataset obtained the ACC of 96.92%, DR of 97.20%, and FPR of 7.49%. The proposed algorithm performance with the NSL-KDD dataset has an ACC of 93.95%, DR of 95.5%, and FPR of 10.34%. However, the main drawback of applying the *k*-NN is that whenever the size of the variables increases, the *k*-NN finds it difficult in predicting the output of the new data positions. On the other side, the *k*-NN does well with the variables having smaller numbers.

Mishra et al. [41] applied the BFS-NB hybrid structure in IDS. This paper proposes the best first search technique for dimensionality reduction which was employed for the attribute selection technique. For the classification of data, Naïve Bayes classifier is applied for a classification purpose and to maximize the accuracy of detecting intrusion. The BFS-NB algorithm is analyzed with the KDD dataset gathered from the US Air Force. The classification accuracy of BS-NFB is 93% while the sensitivity analysis of 97% is achieved. The major disadvantage with the Naive Bayes is that it presumes prediction of the features that are mutually independent to one another.

Dutta et al. [42] introduced a hybrid model for improving the classification metrics in a NIDS. The literature applies a deep neural network for enhancing classification accuracy. Furthermore, classical autoencoder is used as a feature subset selection technique. The efficiency of a proposed technique is evaluated with the UNSW-NB15 dataset. A precision rate of 92.08%, a recall of 90.64%, an accuracy of 91.29%, and *F* -measure of 91.35%, and an FPR of 0.805 are obtained from the proposed architecture. The deep neural network has activation functions and multiple layers that produce nonconvex shapes. The drawback of a deep neural network probably introduces the complex error space, leading to the substantially tuning of hyperparameters to be able to get into a small error space so that the model can be beneficial. Moreover, the training is very slow due to the tuning of many hyperparameters.

Latah and Toker [43] introduce an efficient flow-based multilevel hybrid intrusion detection system. The author applies the k-NN, H-ELM, and ELM which are used for classification purposes, and the SDN controller is used as a feature selection method. An accuracy of 84.29%, FPR of 6.3%, a precision of 94.18%, a recall of 77.18%, and F-measure of 84.83% are obtained from the proposed approach. However, the disadvantages of k-NN are that it is not able to handle well with large and high-dimensional datasets. Furthermore, the k-NN is sensitive to the noise in the dataset.

Sumaiya Thaseen et al. [44] applied the integrated techniques CFS + ANN to improve the classification accuracy. CFS is applied as a feature selection approach for selecting the best feature subsets, while the ANN is employed as a classifier. UNSW-NB15 and NSL-KDD datasets are used for evaluating purposes. An accuracy of 98.45%, a sensitivity of 92.94%, a specificity of 94.38%, and an execution time of 500 seconds are obtained on the NSL-KDD dataset. For the UNSW-NB15 dataset, an accuracy of 96.44%, a sensitivity of 50.4%, specificity of 98.4%, and an execution time of 1023 seconds are achieved. The major disadvantage of ANN is that it takes a longer time for training the data.

Safaldin et al. [45] applied the improved binary gray wolf optimizer as a feature selection method and support vector machine for classification in an IDS in a wireless sensor network. The proposed approach attains an accuracy of 96%, FPR of 0.03, a detection rate of 0.96, and an execution time of 69.6 h. The choosing of a good kernel function is hard which is the major disadvantage of the SVM classifier. Moreover, SVM takes a longer time in training the large datasets, and to store all the support vectors, the memory consumption is extensive.

Vallathan et al. [46] introduce the skeptical action detection system that is based on the deep learning approach in IoT surroundings. Unexpected activities obtained from the footage of the N/W surveillance devices are predicted with the help of deep learning approaches and RFKD. For classification purposes, the multiclassifier approach is used, while DNN is used for training and learning the data. Moreover, for prediction and clustering of data, the kernel density approach is applied. The proposed approach uses the basic merge-sort tree as a feature subset selection approach. For evaluation purposes, HHAR datasets are used. The proposed approach obtained an accuracy rate of 98.4%, specificity of 99.8%, and a sensitivity of 96.02%, on the HHAR dataset. However, the main drawback of the neural network is that the training is very slow due to the tuning of many hyperparameters.

Table 1 depicts the taxonomy of the latest hybrid IDS techniques with its various feature selection approaches. When the literature studies are analyzed, most of them do not classify the dataset according to attack types and attack names thus preventing the assessment of individual attacks on the various networks. Distinct attacks may have peculiar connections as some of the attacks such as R2L and U2R may have very few N/W connections, while other attacks such as Probe and DoS may have a large number of N/W connections or can be a combination of any of them. The attack names found in the attack types help in predicting the vulnerability of individual attacks in various networks. Moreover, a feature selection approach that utilizes highly merit and accurate feature subsets which apply machine learning techniques is not utilized. Furthermore, performance metrics such as precision, MCC, ROC area, PRC area, kappa statistic, MAE, RAE, RMSE, and RRSE which are considered important metrics in model predictability are not utilized in the existing works of literature.

Due to the reviewed problem in the literature studies, a novel hybrid network IDS named NID-Shield has been introduced that employs a distinct machine learning and efficient hybrid feature subset selection approach called CAPPER that is the sequence of the CFS and Wrapper method. Moreover, the hybrid NID-Shield NIDS classifies the dataset according to the various attack names and their types found in the dataset.

2.2. Advantages of Hybrid NIDSs. This section introduces the problem of the existing approaches of IDSs based on anomaly and signature IDSs and explains the advantages of hybrid network intrusion detection systems.

Cybersecurity ventures [47] in the report estimate that the damages arising due to cybercrime in 2025 will increase to $10.5 trillion annually as compared to $3 trillion in 2015. Furthermore, there is a prediction of nearly 7.5 billion active internet users by the end of 2030 worldwide and spending on cybersecurity aggregately surpasses $1 trillion approximately in the coming five years globally.

Despite having enormous financing in the field of IDSs, the losses brought by the intrusions are soaring at an alarming rate leading to enormous debt revenues to the organizations. Considering the efficiency of the IDS, there should be an analytical and stringent proceeding to be acclimated so that network susceptibilities can be classified in a precise and accurate fashion. In past decades, the IDS has been the blocking source for ever-growing intrusion violations and it is utilized as a primary prevention method against computer attacks, safeguarding networks and computer systems. IDS employs statistical techniques, logical operation, and machine learning approaches to analyze distinct kinds of network behaviors [48]. Although present-day IDSs are certainly effective and pursue upgrades, they still develop numerous false alarm rates and fail to analyze the unidentified attacks. Utmost IDSs rely upon inappropriate and redundant inferior level network data to observe cyber intrusions [49]. At two layers of supervision, the existing intrusion detection approaches work to counter the cyberattacks, the host, and the network level. NIDS audits the details of N/W connections to identify the cyberattacks. Contrarily, HIDS scans the workstations' stature and internals of the computing structure utilizing definitive IDS techniques so that at the host level, the potential intrusions can be detected. NIDS is the operating system and platform-independent that does not require any modification when NIDS operates. This makes NIDS more scalable and robust compared with HIDS.

Machine learning analysts classify IDS within three extensive categories: anomaly-based, signature-based, and hybrid-based [50]. The anomaly-based IDS employs the new action profiles which are created every time to distinguish the deviation of outliers from the new profiles. Anomaly-based IDS depends on analytical methods to constitute an attack predictor model. The attack that does not have predefined signatures is recognized by the anomaly-based IDS as its main strength. However, a major weakness lies in the difficulty in creating new action profiles every time. Moreover, the deviations of outliers from the new profiles always are not an attack. Failing to analyze the perimeters of new actions leads to the false prediction of new actions as an attack, possibly ending in a high false-positive rate. The signature-based intrusion detection systems evaluate resemblance among occurrences under scrutiny and the familiar attack patterns. If the patterns formerly established are recognized, then alarms are triggered. For signature-based IDS, e.g., the SNORT [51] is among the utmost preferable, consistently adapted technique. SNORT carries out content seeking, content resembling, and real-time traffic investigation to recognize attacks by employing the predefined precise signatures. Although these systems are definite in analyzing the identified attack, they are incapable to perceive the unidentified attack.

The hybrid-based IDS integrates the anomaly and signature detection approaches to detect attacks. However, the computational expense of utilizing the anomaly and signature IDS that examines the N/W connections is the major drawback of hybrid approaches. The anomaly and signature IDS were the most preferred methods in an organization until various weaknesses were observed leading to the development of hybrid intrusion detection systems. Furthermore, when Table 1 is observed related to hybrid network IDS, most of the literature studies do not classify the dataset according to attack types and their names leading to the difficulty in predicting the attacks individually on different networks. To solve this problem, a novel hybrid NIDS called NID-Shield is proposed in the manuscript that classifies a dataset according to different attack types.

2.3. Machine Learning Algorithms Used in This Study. Distinct machine learning algorithms such as neural network [52], decision trees [53], *k*-nearest neighbor [54], and support vector machine [55] are introduced by the researchers to attain learning on the datasets. Under the contrasting structure of the datasets, the particular algorithms apply distinct methods for achieving higher performance from the datasets. The relevant approach may be applied according to the divergent form of the datasets [56]. Machine learning algorithms such as Naive Bayes, random forest, and J48 (C 4.5) are applied in this study for analyzing the outcome of feature selection and training of the classifier. These algorithms are known to be prominent in the area of machine learning and have proven appropriate in the process.

2.3.1. Random Forest. Random forest [57] is the sequel of tree predictors, and every tree corresponds to the profit of random vector that is sampled independently, and there is an identical distribution of entire trees in the forest. In a forest, as the tree grows larger, the generalization error coincides to a greater extent. The generalization error from a forest relies on the individual strengths of a tree in the forest and correlation between each other in a forest of classifier trees. The random forest performs sequences of inputs or the inputs that are randomly selected at every node so that the accuracy can be increased. By applying this method, the correlation is decreased and simultaneously yields efficacy to forests. The random forest constructs the random features at every node by dividing the limited number from the input variables and electing the features randomly. In the random forest, the tree is grown with the same procedure as the CART [58] approach and the branch that is to be developed is determined by the Gini index. Random forest applies bagging [59] besides the selection of random features. From the standard training dataset, a contemporary training dataset is performed with substitution, and then, on the contemporary training dataset with the help of random feature selection, the tree is grown. Pruning of the tree is not performed on the random forest; rather, the trees are grown in this approach.

Employing bagging has mainly two benefits. Firstly, the accuracy is increased each time the random features are enforced. Secondly, estimation of the generalization error containing the ensemble tree combination and the correlations and its intensity appraisal is provided by the bagging. The assessment is carried out-of-bag [60]. The main approach behind the out-of-bag estimation is the incorporation of nearly one-third of classifiers from the continuing prevailing sequence. Whenever the statistic of the sequence is incremented, the rate of error declines. Therefore, the contemporary error rate can be augmented by out-of-bag estimation; hence, it is necessary to pass on from the area where the merging of the error occurs. In the cross-validation, there is a high probability of the existence of bias; also, the degree of extent of the bias is unfamiliar, whereas the out- of-bag estimation is free from bias. The random forest applies two-thirds of the data and for testing one-third of the data from training data, to grow the tree. Out-of-bag data is simply the one-third data from the training data. Pruning is not performed by the random forest which thus aids in fast and high performance. Moreover, having the multiple tree construction, the random forest performs reasonably well with additional tree framework and it achieves a higher performance rather than any other decision tree method.

2.3.2. Naïve Bayes. Naive Bayes [61] is the classifier having the probabilistic nature, having the relationship relevant to Bayes belief with the strong expectation, and having naive independence between its features. With the kind of probabilistic analysis, the Naive Bayes represent the knowledge. In mathematical terms, the Naive Bayes can be defined as

$$P\left(\frac{R}{S}\right) = P(R)\frac{(S/R)}{P(S)}, \tag{1}$$

where R and S are the events and $P(R)$ and $P(S)$ are the events.

$P(R/S)$ is the posterior probability, having the probability of observation of the event R, given that the S is true. $P(R)$ and $P(S)$ are called the prior probabilities of R and S. $P(S/R$ $)$ is called likelihood, the probability of observation of an event S, given that R is true. The Naive Bayes version that is applied in this study is the implementation by [62]. The nominal feature probabilities are approximated from the

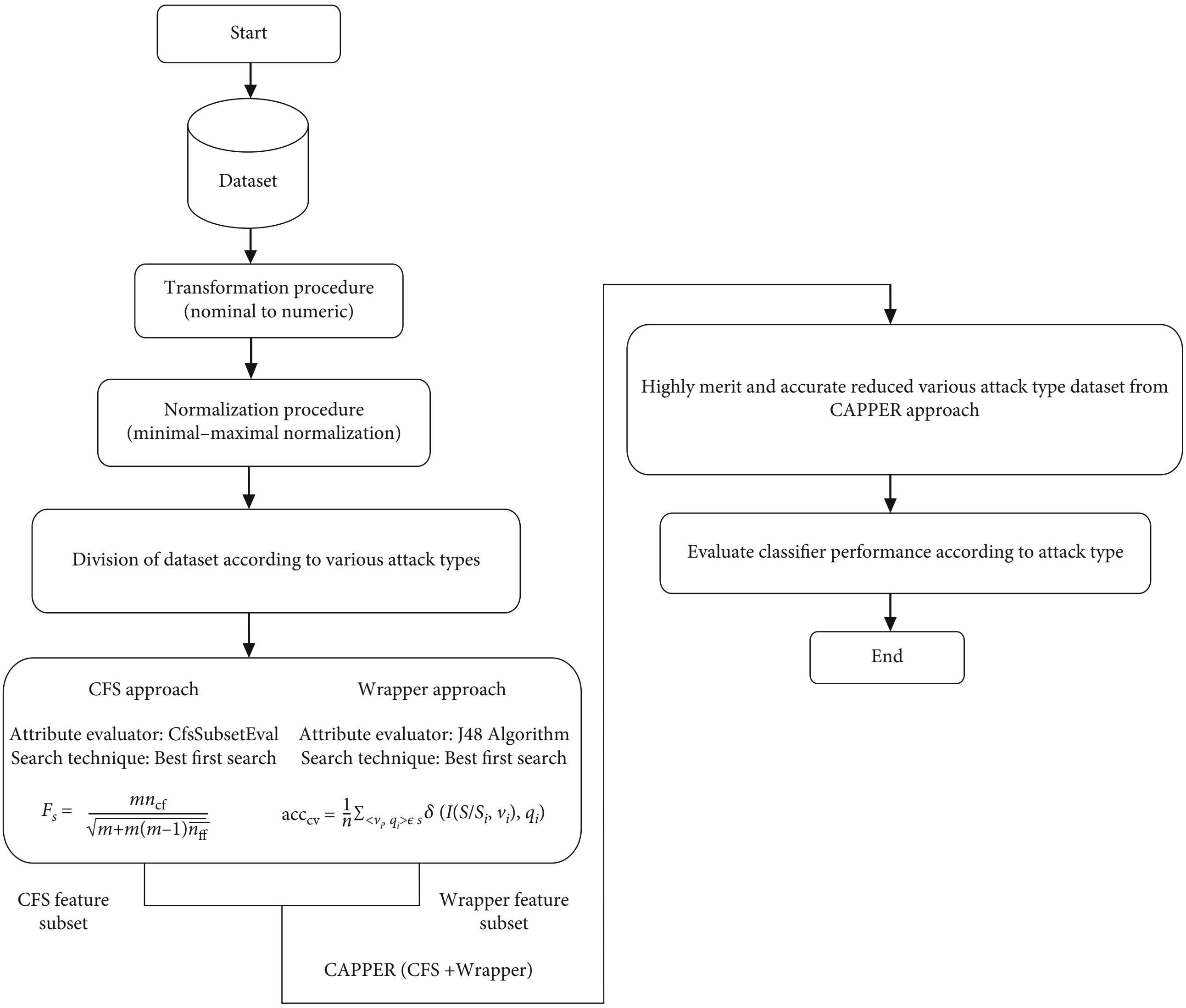

Start
Dataset
Transformation procedure (nominal to numeric)
Normalization procedure (minimal–maximal normalization)
Division of dataset according to various attack types
CFS approach
Wrapper approach
Attribute evaluator: CfsSubsetEval
Search technique: Best first search
Attribute evaluator: J48 Algorithm
Search technique: Best first search
$F_s = \frac{mn_{\mathrm{cf}}}{\sqrt{m+m(m-1)\overline{n_{\mathrm{ff}}}}}$
$\mathrm{acc}_{\mathrm{cv}} = \frac{1}{n}\Sigma_{<v_i,\, q_i>\epsilon\, s}\delta\, (I(S/S_i, v_i), q_i)$
CFS feature subset
Wrapper feature subset
CAPPER (CFS +Wrapper)
Highly merit and accurate reduced various attack type dataset from CAPPER approach
Evaluate classifier performance according to attack type
End

Figure 1: The simplified block diagram of hybrid NID-Shield NIDS according to various attack types.

given data and the Gaussian distribution. The highly apparent class for the given instance based on the entire data distribution is predicted by the Bayes classifier or Bayes rule. Whenever the log probabilities are applied, the Naive Bayes is easy to understand. There are added scoring objectives and natural expression capabilities found in the log probabilities. High accuracy can be obtained from the Bayes classifier. Whenever the redundant features have been eliminated, the performance of the Naive Bayes improves considerably, as discussed by Langley and Sage [63]. Moreover, when modest dependencies prevail in the data, the Naive Bayes performs exceptionally, as discussed by Domingos and Pazzani [64]. A minimal execution time is needed from Naive Bayes to train the data.

2.3.3. J48(C4.5) Decision Tree Generator. C4.5 [65] decision tree is applied in this study. C4.5 is a descendent of an ID3 algorithm. C4.5 is commonly known as J48 in the Weka library. C4.5 constructs the decision trees, and the pruning is performed on the decision trees with the help of the top-down method. The construction of the trees is performed by C4.5 by finding the feature sets having distinct best characteristics so that on the root node of a tree, the testing of the features can be performed. The nodes of the tree relate to its features and branches that relate to its values. The leaf of the tree is reciprocal to the classes, and to classify the new instance, one needs to analyze the features that are tested at the nodes of the tree and pursue the branch corresponding to the values noticed in an instance. The process gets terminated, whenever it arrives at the leaf and also the nomination of the class to its instance.

The greedy approach is used by C4.5 to construct the decision trees which applies the information-theoretic estimates. For obtaining the attribute for tree root, this algorithm splits the instances of the training into subsets which coincides with the attributes corresponding amount. If there is insignificant entropy among the labels of the class in a subset corresponding to labels of the class in the entire training dataset, gaining the information is done by dividing the attribute. The gain ratio principle is enforced by C4.5 for the

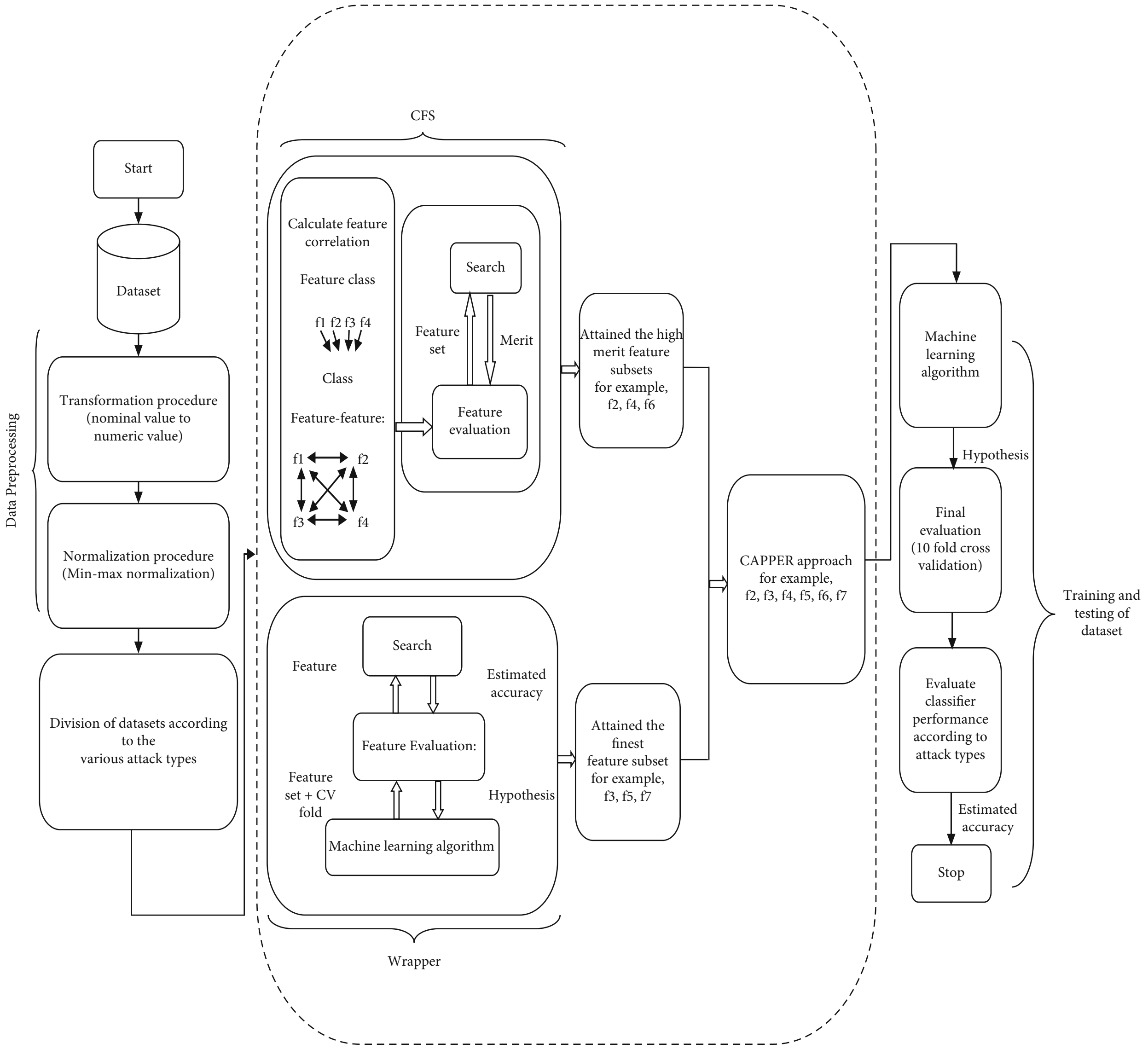

Start
Dataset
Data Preprocessing
Transformation procedure (nominal value to numeric value)
Normalization procedure (Min-max normalization)
Division of datasets according to the various attack types
CFS
Calculate feature correlation
Feature class
f1 f2 f3 f4
Class
Feature-feature:
f1
f2
f3
f4
Feature set
Search
Merit
Feature evaluation
Attained the high merit feature subsets for example, f2, f4, f6
Feature
Search
Estimated accuracy
Feature Evaluation:
Feature set + CV fold
Hypothesis
Machine learning algorithm
Wrapper
Attained the finest feature subset for example, f3, f5, f7
CAPPER approach for example, f2, f3, f4, f5, f6, f7
Machine learning algorithm
Hypothesis
Final evaluation (10 fold cross validation)
Evaluate classifier performance according to attack types
Estimated accuracy
Stop
Training and testing of dataset

Figure 2: A proposed architecture of hybrid NID-Shield network intrusion detection system

Table 2: Four categories of attack.

Attack category	Name of attack
Denial of service (DoS)	teardrop, smurf, neptune, back, land, pod
Probe	satan, nmap, ipsweep, portsweep
User to root (U2R)	loadmodule, buffer_overflow, rootkit
Remote to local (R2L)	multihop, phf, ftp_write, warezclient, imap, guess_passwd, warezmaster

Table 3: Number of instances in NSL-KDD and NSL-KDD 20% training on normal and attack type.

NSL-KDD dataset	Normal	Probe	DoS	U2R	R2L	Total instances
NSL-KDD training	67343	11656	45927	52	995	125973
NSL-KDD 20% training	13449	2289	9234	11	209	25192

Table 4: Features of NSL-KDD 20% dataset.

Index	Feature name	Type	Missing	Distinct	Unique	Mean	Std. Dev.
1	duration	Numeric	0	758	682	0.007	0.063
2	protocol_type	Nominal	0	3	0	0.125	0.283
3	service	Nominal	0	66	1	0.157	0.198
4	flag	Nominal	0	11	0	0.062	0.103
5	src_bytes	Numeric	0	1665	864	0	0.006
6	dst_bytes	Numeric	0	3922	2377	0.001	0.017
7	land	Nominal	0	2	0	0	0.009
8	wrong_fragment	Numeric	0	3	0	0.008	0.087
9	urgent	Numeric	0	2	1	0	0.006
10	hot	Numeric	0	22	7	0.003	0.028
11	num_failed_logins	Numeric	0	5	2	0	0.011
12	logged_in	Numeric	0	2	0	0.395	0.489
13	num_compromised	Numeric	0	28	18	0	0.012
14	root_shell	Numeric	0	2	0	0.002	0.039
15	su_attempted	Numeric	0	3	0	0.001	0.024
15	num_root	Numeric	0	28	20	0	0.012
17	num_file_creations	Numeric	0	20	13	0	0.013
18	num_shells	Numeric	0	2	0	0	0.019
19	num_access_files	Numeric	0	7	2	0.001	0.012
20	num_outbound_cmds	Numeric	0	1	0	0	0
21	is_host_login	Nominal	0	1	0	0	0
22	is_guest_login	Numeric	0	2	0	0.009	0.095
23	count	Numeric	0	466	70	0.164	0.225
24	srv_count	Numeric	0	414	69	0.052	0.142
25	serror_rate	Numeric	0	70	9	0.286	0.447
26	srv_serror_rate	Numeric	0	56	25	0.284	0.448
27	rerror_rate	Numeric	0	72	9	0.119	0.319
28	srv_rerror_rate	Numeric	0	42	10	0.12	0.322
29	same_srv_rate	Numeric	0	97	7	0.661	0.44
30	diff_srv_rate	Numeric	0	79	14	0.062	0.179
31	srv_diff_host_rate	Numeric	0	57	4	0.096	0.257
32	dst_host_count	Numeric	0	256	1	0.716	0.388
33	dst_host_srv_count	Numeric	0	256	1	0.451	0.434
34	dst_host_same_srv_rate	Numeric	0	101	0	0.52	0.449
35	dst_host_diff_srv_rate	Numeric	0	101	0	0.083	0.187
36	dst_host_same_src_port_rate	Numeric	0	101	0	0.147	0.308
37	dst_host_srv_diff_host_rate	Numeric	0	63	8	0.032	0.111
38	dst_host_serror_rate	Numeric	0	100	5	0.286	0.445
39	dst_host_srv_serror_rate	Numeric	0	88	19	0.28	0.446
40	dst_host_rerror_rate	Numeric	0	101	0	0.118	0.306
41	dst_host_srv_rerror_rate	Numeric	0	100	7	0.119	0.317

selection tree of the root attribute. The gain ratio principle elects those attributes which have an average or better gain between its distinct attributes.

By employing the algorithm iteratively, subtrees are constructed in this algorithm. Furthermore, the algorithm terminates upon finding the likely subset that contains a distinct class. The main distinction between C4.5 and ID3 is that pruning is performed on decision trees by C4.5; hence, by applying the pruning, the simplification is done on the decision trees and has the high chance of reducing the overfitting on a training data. C4.5 performs pruning by employing the confidence interval upper bound on the resubstitution error. The succession of the node is preceded by the best leaf, whenever the error of the estimation of the leaf is situated within a single standard deviation from the predicted error of a node. C4.5 is considered as an efficient algorithm, whenever the efficacy regarding the machine learning algorithm is assessed; also, it is fast, robust, and accurate whenever the knowledge is brought in. Moreover, it performs well with feature subset selection on the relevant and redundant data, thus aiding in increasing the accuracy.

3. The Proposed Hybrid NID-Shield Network Intrusion Detection System

This section introduces the various techniques applied by the hybrid NID-Shield NIDS. The data preprocessing steps are performed by applying the transformation and normalization operations on datasets, and then, an effective hybrid feature subset selection technique called CAPPER is applied for obtaining the accurate and highest merit feature subsets. Finally, the hybrid NID-Shield NIDS is suggested as a whole exclusively.

3.1. Data Preprocessing. In data preprocessing, the transformation and normalization operation is performed on NSL-KDD 20% dataset. It can help to better expose the underlying structure of the data to the learning algorithm and, in turn, may result in better predictive performance.

3.1.1. Data Transformation. In the transformation operation, the nominal values are converted to numeric values. The IDSs are considered as the classification issue and some classification approaches are not able to handle the nominal features [66]. In the NSL-KDD 20% dataset, the attributes such as protocol_type, service, and flag are transformed from nominal to numeric values and the final NSL-KDD 20% dataset contains the entire numeric values for the classification process.

3.1.2. Data Normalization. Data normalization is an essential paradigm, specifically in the area of classification. The instances are observed as a multidimensional area in the linear classification approaches. Without normalization, few objective functions do not work accordingly due to the wide variations of raw data. For example, if the particular feature has wide value ranges, then the range within the points is controlled by the distinct feature. Thus, normalization of the numeric features needs to be done so that every feature provides nearly proportional to the eventual distance. Therefore, by applying the normalization, there is a significant improvement in accuracy and speed. For this study, minimal-maximal normalization approach is applied to the dataset. The minimal-maximal normalization is given as

$$z = \frac{x - \text{minimal}(x)}{[\text{maximal}(x) - \text{minimal}\,(x)]}. \tag{2}$$

The minimal-maximal normalization technique linearly scales each feature to the interval of [0, 1]. Resizing of the interval [0, 1] is performed by altering every feature value such that the minimum value is 0, and then, division is performed by the current maximum value. The current maximum value is the change among the initial maximum value and minimum value which is obtained from equation (2).

Table 5: Total instances in UNSW-NB 15 training and testing dataset.

S.no.	Total instances	Training dataset (UNSW-NB15)	Testing dataset (UNSW-NB15)
1	Normal	56000	37000
2	DoS	12264	4089
3	Fuzzers	18184	6062
4	Analysis	2000	677
5	Worms	130	44
6	Exploits	33393	11132
7	Shellcode	1133	378
8	Generic	40000	18871
9	Reconnaissance	10491	3496
10	Backdoor	1746	583

3.2. Feature Selection Approaches. A hybrid feature subset selection approach named CAPPER [67] is employed for feature subset selection that combines the feature subsets from the CFS and Wrappers for the feature subset selection method. This section introduces the CAPPER approach.

3.2.1. Correlation-Based Feature Subset Approach. CFS is the filter method that utilizes correlation-based searching for the appraisal of the feature subsets. The feature subset ranking is accomplished by conferring to correlation-based searching. The bias is accessed to those subsets which are greatly correlated to its class and uncorrelated among them. This approach ignores the features that are irrelevant and having fewer correlations among its class. The screening is applied to the features which are redundant and hugely correlated among its class. The acceptance of the features is done by the CFS when the residual features do not predict the predicted class in the instant space.

$$F_S = \frac{m\overline{n_{cf}}}{\sqrt{m + m(m-1)\overline{n_{ff}}}}, \tag{3}$$

where F_S is the heuristic merit of the feature subset S having the m features, $\overline{n_{\text{ff}}}$ is the average feature-feature intercorrelation, and $\overline{n_{\text{cf}}}$ are the feature mean class correlation. The searching of the space is performed with the help of a best-first approach. The high-quality subset of the features is obtained by equation (3), which aids in reducing the dimensional reduction of testing and training data. Moreover, the numerator of equation (3) illustrates that how remarkably the class predictability is with feature sets and the denominator denotes the redundancy between the features.

3.2.2. Wrapper Subset Selection Approach. In the Wrapper approach, the feature subset selector is performed with the help of an induction approach. The searching of the feature subset space is performed with the help of backward elimination and forward selection methods. The backward elimination begins with complete feature sets and removing those features that degrade the performance. The forward selection

Table 6: Features of UNSW-NB 15 dataset.

Index	Feature name	Type	Missing	Distinct	Unique	Mean	Std. Dev.
1	id	Numeric	0	82332	82332	0.5	0.289
2	dur	Numeric	0	39888	35946	0.017	0.079
3	proto	Nominal	0	131	0	0	0.002
4	service	Nominal	0	13	0	0.26	0.438
5	state	Nominal	0	7	2	0.047	0.5
6	spkts	Numeric	0	420	174	0.002	0.013
7	dpkts	Numeric	0	436	205	0.002	0.01
8	sbytes	Numeric	0	4489	2570	0.001	0.012
9	dbytes	Numeric	0	4034	2446	0.001	0.01
10	rate	Numeric	0	40616	32279	0.082	0.149
11	sttl	Numeric	0	11	1	0.71	0.398
12	dttl	Numeric	0	8	0	0.378	0.461
13	sload	Numeric	0	42873	38993	0.012	0.034
14	dload	Numeric	0	40614	37491	0.03	0.115
15	sloss	Numeric	0	253	101	0.001	0.012
15	dloss	Numeric	0	311	124	0.001	0.01
17	sinpkt	Numeric	0	39970	36718	0.013	0.103
18	dinpkt	Numeric	0	37617	34993	0.002	0.022
19	sjit	Numeric	0	39944	37503	0.004	0.038
20	djit	Numeric	0	38381	36358	0.001	0.008
21	swin	Nominal	0	11	9	0.523	0.499
22	stcpb	Numeric	0	39219	37322	0.253	0.324
23	dtcpb	Numeric	0	39108	37295	0.25	0.322
24	dwin	Numeric	0	14	11	0.503	0.5
25	tcprtt	Numeric	0	26130	22613	0.015	0.03
26	synack	Numeric	0	24934	20749	0.009	0.022
27	ackdat	Numeric	0	24020	19622	0.009	0.019
28	smean	Numeric	0	1282	178	0.078	0.141
29	dmean	Numeric	0	1222	236	0.078	0.163
30	trans_depth	Numeric	0	8	4	0.001	0.004
31	response_body_len	Numeric	0	1190	809	0	0.007
32	ct_srv_src	Numeric	0	57	0	0.138	0.179
33	ct_state_ttl	Numeric	0	7	1	0.228	0.178
34	ct_dt_ltm	Numeric	0	50	1	0.082	0.145
35	ct_src_dport_ltm	Numeric	0	50	1	0.068	0.145
36	ct_dst_sport_ltm	Numeric	0	33	1	0.072	0.16
37	ct_dst_src_ltm	Numeric	0	57	0	0.104	0.184
38	is_ftp_login	Numeric	0	3	0	0.004	0.046
39	ct_ftp_cmd	Numeric	0	3	0	0.004	0.046
40	ct_ftw_http_mthd	Numeric	0	8	0	0.008	0.04
41	ct_src_ltm	Numeric	0	50	1	0.093	0.145
42	ct_srv_dst	Numeric	0	57	0	0.134	0.182
43	is_sm_ips_ports	Numeric	0	2	0	0.011	0.105
44	attack_cat	Nominal	0	10	0	0.074	0.261
45	label	Numeric	0	2	0	0.551	0.497

begins with empty feature sets and starts adding the good features. The goal of this approach is to obtain the state with the highest appraisal by applying the heuristic function. For the appraisal function, the five fold cross-validation approach is performed, and it is replicated numerous times by examining the accuracy estimation and its standard deviation. The *k*

-fold cross-validation is also called an out-of-sample test or rotation estimation. S is the original sample, which splits into folds of $S_1, S_2, S_3,\ldots,S_n$ relatively identical size every time $t \in \{1, 2,. \cdots, k\}$ trained on $S \setminus S_t$ and tested on S_t. The induction approach is tested and trained k times. The estimation of the cross-validation accuracy is the comprehensive figure of accurate classifications divided from the total instances from the dataset. Let S_i is the testing set that contains the instances $p_i = <v_i, q_i>$; then, the estimation of cross-validation accuracy is obtained as

$$\mathrm{acc}_{cv} = \frac{1}{n} \sum_{<v_i,q_i> \in S} \delta(I(S \setminus S_i, v_i), q_i). \tag{4}$$

The best-first approach is applied as the search technique. Upon arriving at the goal, the best-first search usually terminates. The accuracy estimation is obtained from equation (4). By combining the feature subsets from Wrapper and CFS approaches, CAPPER attains the accurate and high-quality feature subsets.

3.3. *Ensemble Learning.*

Ensemble learning [68] was initially evolved in automated decision-making systems to lessen the variance and thus increase the accuracy. The problems in machine learning domains such as error correction, estimation confidence, missing features, and cumulative learning are strongly addressed by ensemble learning techniques. Ensemble learning is widely used in the area of pattern recognition, artificial intelligence, machine learning, data mining, and neural networks. Ensemble learning has proved its efficiency and functionality in an extensive area of real-world problems.

The ensemble learning combines various base learners or weak learners and integrates them to make a strong learner. The superiority of ensemble learning is that it increases the accuracy of the weak learning system so that the comprehensive accuracy of the classifier on the training datasets is increased as compared to the single base learning algorithms.

3.3.1. *Stacking.*

In stacking [69], the cardinal classifier obtains a new dataset from the original datasets. If the same instances are generated from the original dataset by the cardinal classifier, then there is high speculation that the data gets overfitted, which is the primary reason the datasets with contemplating nature need to be obtained for discarding the overfitting of the data. There is a suggestion to use the cross-validation approach for the new instances of the cardinal classifier; also, the group of features has to be considered for the contemporary training dataset and the different categories of the learning algorithms on the Meta-learner. Distinct learning algorithms are applied for obtaining the cardinal learner. Then, the new datasets are used with Meta-learner to train the data. Stacking is the induction of numerous machine learning approaches.

3.4. *k-Fold Cross-Validation.*

Cross-validation techniques are frequently mentioned as test/train holdout approach by the researchers. In the k-fold cross-validation [70], the repetition on the dataset is performed k times. At every round, the dataset is split into k parts; one part is applied for the validation and the residual k-1 parts of the datasets are combined into a training subset for appraisal of the model. In k-fold cross-validation, a complete set of testing and training data is used, and the main idea of this technique is to lessen the fatalistic bias by applying the major number of training data while keeping the large testing datasets separately. The folds of the test data do not overlap each other. In k-fold cross-validation, each of the samples is applied for validation. Sometimes, it is necessary to choose the exact value of k to avoid the high bias in the model. Usually, the value of $k = 10$ is chosen mostly, as the various experimental results show that the model has small bias and low variance whenever this value is applied. The results from this approach are then combined or averaged to generate the distinct estimation.

Table 7: Confusion matrix.

Class	Predicted negative class	Predicted positive class
Actual negative class	FP	TN
Actual positive class	TP	FN

3.5. *The Proposed Hybrid NID-Shield Network Intrusion Detection System Using Hybrid Feature Selector.*

The preliminary design approach behind the hybrid NID-Shield is the classification of datasets according to different attack types. The advantage of classification of the dataset according to attack types is that it can find a set of arbitrary features. Moreover, the attack names found in the attack types help in predicting the vulnerability of individual attacks in various networks. Distinct machine learning algorithms are analyzed as per the individual attack types. The machine learning algorithms having high accuracy; low FPR are selected for different attack types and applied in the designing of the hybrid NID-Shield NIDS. The hybrid NID-Shield NIDS applies the hybrid approach called CAPPER for selecting the optimal feature subsets. The hybrid CAPPER approach for feature selection combines the optimal feature subsets from the CFS and Wrappers for the feature subset selection method. From the CFS approach, a prominently superior feature subset is obtained which is independent of irrelevant and redundant features. The wrapper method uses induction learning algorithms to attain a highly accurate feature subset. By combining the filter and wrapper approaches, high merit and accurate feature subsets are obtained which is then applied for training and testing purposes.

For designing the hybrid NID-Shield NIDS, single and ensemble learning algorithms are used together so that a high-performance rate and lower FPR can be achieved. Testing is performed with single and ensemble learning algorithms; it has been found that ensemble learning achieved high-performance results, where the NSL–KDD 20% having fewer samples in some of its attack types. The high-performance classifier is determined for different attack types, and for the classifier performance, the k-fold cross-

Table 8: DOS attack evaluated with hybrid NID-Shield NIDS approach.

(a)

Total instances	22,683
Correctly classified instances	22,679 (99.98%)
Incorrectly classified instances	4 (0.00176%)
Execution time	6.77 seconds
Kappa measures	0.9997
MAE	0.0003
RMSE	0.0081
RAE	0.2207%
RRSE	2.9778%

(b)

Accuracy		TP rate	FP rate	Precision	Recall	*F*-measure	MCC	ROC area	PRC area	Class
100%		1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	normal
100%		1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	back
100%		1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	land
100%		1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	neptune
94.7%		0.947	0.000	0.973	0.947	0.960	0.960	1.000	0.997	pod
99.6%		0.996	0.000	0.998	0.996	0.997	0.997	1.000	1.000	smurf
100%		1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	teardrop
Weighted Avg.	100%	1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	

(c)

			Confusion matrix			
a	*b*	*c*	*d*	*e*	*f*	*g*
13,449	0	0	0	0	0	0
0	196	0	0	0	0	0
0	0	4	0	0	0	0
0	0	0	8279	0	0	0
0	0	1	0	36	1	0
0	0	0	0	2	527	0
0	0	0	0	0	0	188

a—classified as normal, *b*—classified as back, *c*—classified as land, *d*—classified as neptune, *e*—classified as pod, *f*—classified as smurf, *g*—classified as teardrop.

validation approach is applied. The advantage of this method is that all observations are used for training and validation, and each observation is used for validation exactly once. For the classification problem, a cross-validation technique with 10-fold is applied. The folds are selected in a manner such that every fold consists of the approximately identical distribution of the class. To test the network data according to attack type, the various attack type data are passed to different layers of the hybrid NID-Shield NIDS; the high-performance classifier determines the data as normal or attack type. Figure 1 depicts the simple block diagram of the hybrid NID-Shield network intrusion detection system and Figure 2 displays the architecture of the hybrid NID-Shield NIDS.

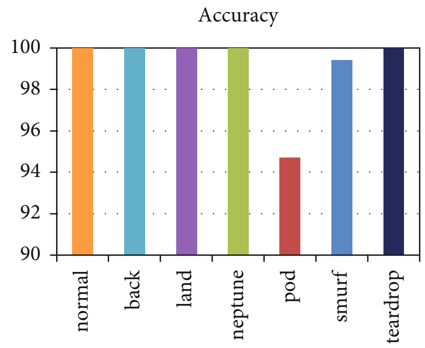


Figure 3: Accuracy of the normal and attack types evaluated by the NID-Shield NIDS on DoS attack.

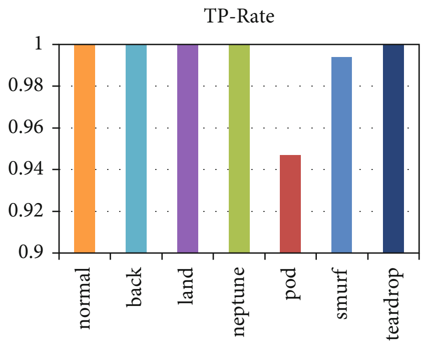


Figure 4: TP rate of the normal and attack types evaluated by the NID-Shield NIDS on DoS attack.

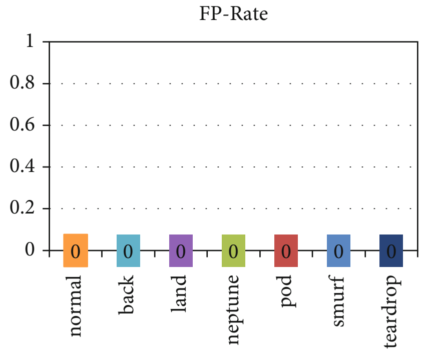


Figure 5: FP rate of the normal and attack types evaluated by the NID-Shield NIDS on DoS attack.

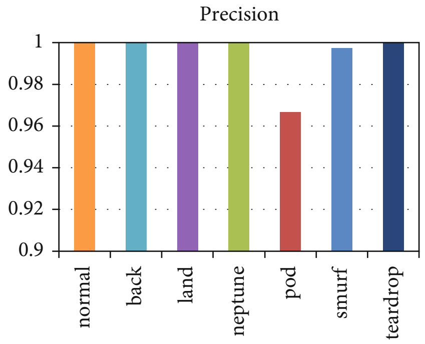


Figure 6: Precision of the normal and attack types evaluated by the NID-Shield NIDS on DoS attack.

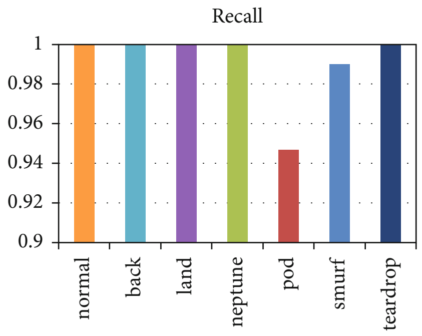


Figure 7: Recall of the normal and attack types evaluated by the NID-Shield NIDS on DoS attack.

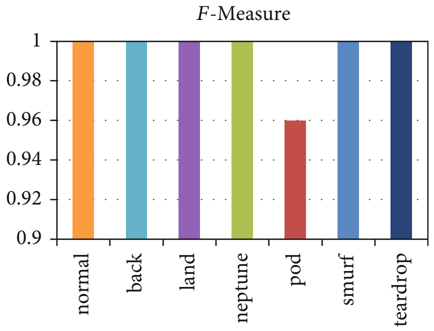


Figure 8: F-measure of the normal and attack types evaluated by the NID-Shield NIDS on DoS attack.

4. Dataset Characteristics

For the performance of the hybrid NID-Shield NIDS, two contemporary UNSWNB-15 and NSL-KDD 20% datasets are utilized for evaluation purposes. These datasets are related to cybersecurity and are high-dimensional and class imbalanced datasets [71]. For the NSL-KDD dataset, the statistical prevalence of around 36% was found in denial of service (DoS), while for other attack types like Root to Local (R2L) and User to Root (U2R), the prevalence is lesser than 1%. This shows that NSL-KDD is a highly imbalanced dataset. For the UNSW-NB15 dataset, the normal class frequency is about 32%, while attack type frequency is very few and differs highly. For example, Worms and Exploits attack patterns vary around 257 times. This reflects that UNSW-NB15 is a highly imbalanced dataset.

4.1. NSL-KDD Dataset. From DARPA 98 intrusion detection system appraisal programs, the KDD-Cup ‘99 dataset is obtained and widely applied dataset in the domain of IDS, but the main disadvantage of the KDD-Cup ‘99 datasets has various duplicate and redundant records. The duplicate records have a total of 75%. The redundant record has a total of 78%. Due to this duplication and redundant information hinders from categorizing the additional records [72]. A new NSL-KDD dataset was suggested [73] that does not contain the duplicate and redundant records in testing and training data [74], which aided in removing the duplicate and redundant issues which is an implicit issue in KDD-Cup ‘99 dataset. The arrangements of elected records from every adversity class level are inversely proportional to the percent of records available in the standard KDD datasets. With these results, the classifying rates of apparent machine learning approaches differ in an extensive range that makes it more efficient to obtain a precise appraisal of distinct learning approaches. The statistical records in the training and testing sets are feasible that causes it to be reasonable to conduct the experiments on an entire set, thus preventing the unnecessary need to randomly select the limited part. Therefore,

Table 9: Probe attack evaluated with hybrid NID-Shield NIDS approach without stacking.

(a)

Total instances	15,738
Correctly classified instances	15,685
Incorrectly classified instances	53
Execution time	4.23 seconds
Kappa measures	0.9872
MAE	0.0034
RMSE	0.0343
RAE	3.1818%
RRSE	14.9173%

(b)

Accuracy		TP rate	FP rate	Precision	Recall	*F*-measure	MCC	ROC area	PRC area	Class
99.9%		0.999	0.016	0.997	0.999	0.998	0.988	1.000	1.000	normal
99.3%		0.993	0.000	1.000	0.993	0.997	0.996	0.999	0.998	portsweep
96.8%		0.968	0.000	0.999	0.968	0.983	0.982	0.999	0.996	satan
99.3%		0.993	0.001	0.989	0.993	0.991	0.990	1.000	0.996	ipsweep
95.3%		0.953	0.000	0.976	0.953	0.965	0.964	0.998	0.992	nmap
Weighted Avg.	99.7%	0.997	0.014	0.997	0.997	0.997	0.988	1.000	0.999	

(c)

		Confusion matrix		
a	*b*	*c*	*d*	*e*
13441	0	1	0	3
2	593	0	0	0
4	0	679	1	2
2	0	0	718	2
0	0	0	3	287

a—classified as normal, *b*—classified as portsweep, *c*—classified as satan, *d*—classified as ipsweep, and *e*—classified as nmap.

the classified results will be persistent and proportionate. There are entirely 37 attacks in a testing dataset out of which 21 different attacks are of training dataset and the remaining attacks are available only for testing the data. Table 2 shows the four categories of attack. Table 3 depicts the total number of instances on the distinct attack types and normal and on the NSL-KDD and 20% of the NSL-KDD training dataset. The attack classes are categorized into Probe, DoS, U2R, and R2L categories, and Table 4 shows the features of the NSL-KDD 20% dataset.

(i) *Denial of service (DoS)*. These kinds of attack result in the unavailability of computing resources to legitimate users. The intruder overloads the resources, by accomplishing the resources of the computer active, so that authentic users are unable to utilize the full resources of the computer. In DoS, there are 13449 normal instances and 9234 attack instances with six attack names, namely, neptune, smurf, back, teardrop, pod, and land

(ii) *Probe*. The intruder gathers the knowledge from the networks or hosts and scans the whole networks or hosts that are prone to attacks. An intruder then exploits the system vulnerabilities by looking at the known security breaches so that the whole system is compromised for malicious purposes. In Probe, there are 13449 normal instances and 2289 attack instances with four attack names, namely, nmap, ipsweep, satan, and portsweep

(iii) *User to root (U2R)*. An intruder tries to acquire accessing the system roots or the administrator privileges by sniffing the passwords. The attacker then looks for the vulnerabilities in the system, to acquire the gain of the administrator authorization. In U2R, there are 13449 normal instances and 11 attack instances with three attack names, namely, loadmodule, buffer_overflow, and rootkit

(iv) *Root to local (R2L)*. The intruder attempts by gaining a connection to the remote machine, which does not

Table 10: Probe attack evaluated with hybrid NID-Shield NIDS approach with stacking.

(a)

Total instances	15,738
Correctly classified instances	15,690
Incorrectly classified instances	48
Execution time	42.99
Kappa measures	0.9884
MAE	0.002
RMSE	0.0318
RAE	1.8696%
RRSE	13.82%

(b)

Accuracy		TP rate	FP rate	Precision	Recall	*F*-measure	MCC	ROC area	PRC area	Class
99.9%		0.999	0.010	0.998	0.999	0.999	0.990	0.999	1.000	normal
99.7%		0.997	0.000	1.000	0.997	0.998	0.998	1.000	1.000	portsweep
97.7%		0.977	0.000	0.993	0.977	0.985	0.984	0.995	0.990	satan
99.3%		0.993	0.001	0.987	0.993	0.990	0.990	0.999	0.994	ipsweep
96.3%		0.963	0.001	0.967	0.963	0.965	0.964	0.997	0.986	nmap
Weighted Avg.	99.7%	0.997	0.009	0.997	0.997	0.997	0.990	0.999	0.999	

(c)

Confusion matrix				
a	*b*	*c*	*d*	*e*
13435	0	5	5	4
2	585	0	0	0
8	2	675	1	2
1	2	0	705	4
3	2	4	3	290

a—classified as normal, *b*—classified as portsweep, *c*—classified as satan, *d*—classified as ipsweep, and *e*—classified as nmap.

have the necessary and legal privilege to access that machine. The attacker then exploits the susceptibility of the remote system and tries gaining access rights to the remote machine. There are 13449 normal instances and 209 attack instances in this dataset. There are eight attack names in this dataset, namely, ftp_write, guess_passwd, multihop, phf, imap, warezclient, spy, and warezmaster

4.2. UNSW-NB15 Dataset. The UNSW-NB15 [75] dataset was generated at the cyber range lab by the IXIA PerfectStorm tool at the Center for cybersecurity, Australia. There are 2,540,044 records in the dataset. The part of the dataset is further divided into train and test sets. There are 82,332 records in the testing set and 1,75,341 records in the training set, having normal and attack instances. There are 45 features in this dataset obtained in immaculate format, including class and label. Moreover, there are nine attack types in a UNSW-NB15 dataset: DoS, Analysis, Backdoor, Exploit, Fuzzers, Generic, Worm, Shellcode, and Reconnaissance and a Normal instance. Table 5 shows the total instances in UNSW-NB 15 training and testing dataset, and Table 6 depicts the UNSW-NB 15 dataset and its features.

For the evaluation of the proposed approach, the machine learning workbench tool, Weka 3.8 [76], is used. In Weka, the Wrapper approach, the CFS approach, and the classifier algorithms like J48, Naïve Bayes, and Random forest are implemented in Java, and evaluation of code is accomplished on Intel i3 8100 processor with 2.20 GHz having 4.00 GB RAM and carried out on NetBeans 8.0.2.

5. Performance Metrics

For validation of the results, this section presents various performance evaluation metrics. The researchers apply false negative (FN), true negative (TN), true positive (TP), false positive (FP), etc. [77] for the justification of the results.

Definition 1 (confusion matrix). Also called error metric, which allows the interaction among actual and predicted classes. It is significant for calculating precision, recall, accuracy, specificity, AUC, and ROC curve. On the testing

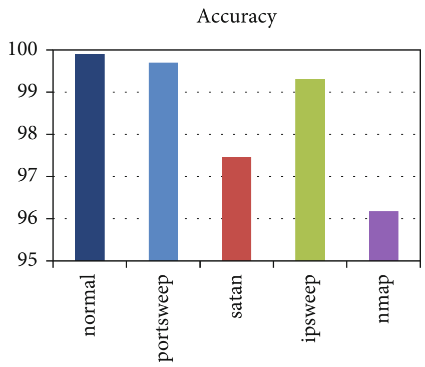


Figure 9: Accuracy of the normal and attack types evaluated by the NID-Shield NIDS on Probe attack.

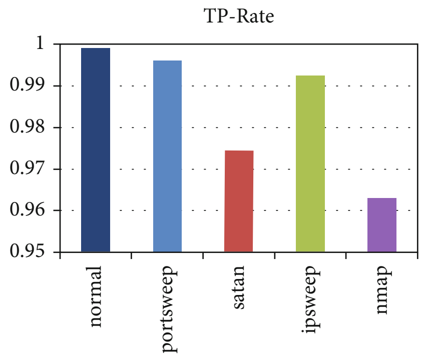


Figure 10: TP-Rate of the normal and attack types evaluated by the NID-Shield NIDS on Probe attack.

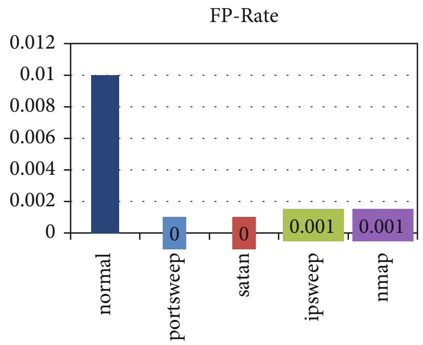


Figure 11: FP-Rate of the normal and attack types evaluated by the NID-Shield NIDS on Probe attack.

dataset, the confusion matrix allows visualizing the algorithms' efficiency and is usually adapted to describe the classifier performance. Table 7 shows the confusion matrix.

Definition 2 (accuracy). The proportion of correct predictions of calculating the classification instances precisely is obtained from

$$\mathrm{acc} = \frac{\mathrm{TP} + \mathrm{TN}}{(\mathrm{TN} + \mathrm{FN} + \mathrm{TP} + \mathrm{FP})}. \tag{5}$$

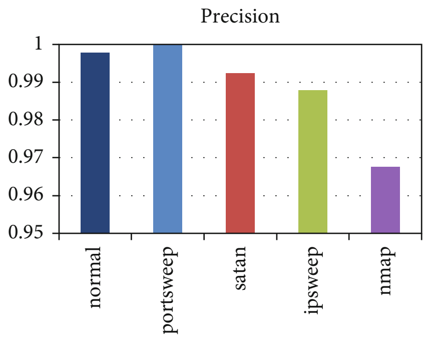


Figure 12: Precision of the normal and attack types evaluated by the NID-Shield NIDS on Probe attack.

Definition 3 (error rate). The proportion of whole predictions done that are classified falsely: it is given by

$$\mathrm{ERR} = 1 - \mathrm{acc}. \tag{6}$$

Definition 4 (true positive). The intrusions are accurately classified as an attack by the intrusion detection systems. It is also called sensitivity, recall, or detection rate. It is obtained from

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}. \tag{7}$$

Definition 5 (false positive). The usual patterns which are misclassified as attacks and calculated as

$$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}}. \tag{8}$$

Definition 6 (true negative). The usual patterns that are precisely analyzed as normal and obtained from

$$\mathrm{TNR} = 1 - \mathrm{FPR}. \tag{9}$$

Definition 7 (false negative). The intrusions misclassified as normal and obtained from

$$\mathrm{FNR} = 1 - \mathrm{TPR}. \tag{10}$$

Definition 8 (precision). The behaviors that are exactly arrayed as attacks and given by

$$\mathrm{Precision} = \frac{\mathrm{TP}}{\mathrm{FP} + \mathrm{TP}}. \tag{11}$$

Definition 9 (*F*-measure). It is interpreted as the harmonic mean of recall and precision. Also known as *F*-score or *F* -value and calculated as

$$\mathrm{FM} = 2 \times \frac{\text{precision x recall}}{\text{precision} + \text{recall}}. \tag{12}$$

Definition 10 (Matthews's correlation coefficient). Applied only in the binary intrusion detection system in which it computes the observed and predicted values of binary classification. It is calculated by

$$\mathrm{MCC} = \frac{(\mathrm{TPxTN}) - (\mathrm{FPxFN})}{\sqrt{(\mathrm{TP}+\mathrm{FP})(\mathrm{TP}+\mathrm{FN})(\mathrm{TN}+\mathrm{FP})(\mathrm{TN}+\mathrm{FN})}}. \tag{13}$$

Definition 11 (kappa statistic). It is applied to calculate the concurrence between observed and predicted values of the datasets, while the concurrence is corrected that occurs unexpectedly. It is calculated by

$$k = \frac{p_0 - p_e}{1 - p_e}, \tag{14}$$

where p_0 is the comparative noticed concurrence between the estimates and p_e is the assumed likelihood of possible concurrence.

Definition 12 (mean absolute error). It is the averaging of the magnitude of the distinctive error and the computing the standard of absolute errors. It is calculated as

$$\mathrm{MAE} = \frac{|p_1 - a_1| + \cdots + |p_n - a_n|}{n}, \tag{15}$$

where p_1 is the value predicted on the test instances and a_1 is the actual value.

Definition 13 (root mean-squared error). The RMSE calculates the dissimilarities among observed values and predicted values of a model. It is given by

$$\mathrm{RMSE} = \sqrt{\frac{(p_1 - a_1)^2 + \cdots + (p_n - a_n)^2}{n}}, \tag{16}$$

where p_1 is the value predicted on the test instances and a_1 is the actual value.

Definition 14 (relative absolute error). The errors are normalized from the errors of simple predictors in which the average value is predicted. It is calculated as

$$\mathrm{RAE} = \frac{|p_1 - a_1| + \cdots + |p_n - a_n|}{|a_1 - \bar{a}| + \cdots + |a_n - \bar{a}|}, \tag{17}$$

where p_1 is the value predicted on the test instances and a_1 is the actual value.

Definition 15 (root relative squared error). It normalizes the total squared error by division of the total squared error from the simple predictor. It is obtained from

$$\mathrm{RRSE} = \sqrt{\frac{(p^1 - a^1)^2 + \cdots + (p_n - a_n)^2}{(a^1 - \bar{a})^2 + \cdots + (a_n - \bar{a})^2}}, \tag{18}$$

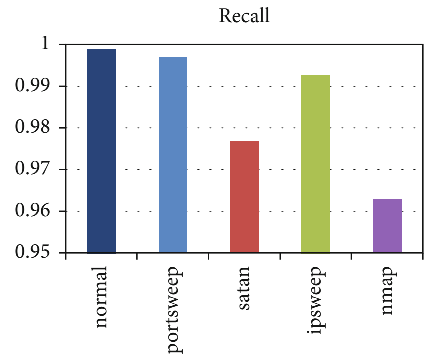


Figure 13: Recall of the normal and attack types evaluated by the NID-Shield NIDS on Probe attack.

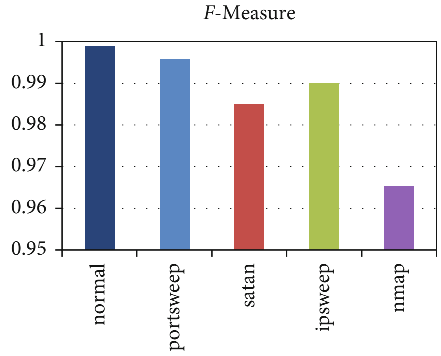


Figure 14: *F*-measure of the normal and attack types evaluated by the NID-Shield NIDS on Probe attack.

where p_1 is the value predicted on the test instances and a_1 is the actual value.

Definition 16 (AUC and ROC). ROC explains detection ratio changes in contrast to its internal verge to develop a high or low FPR. The larger the AUC values, the better the performance of the classifier.

5.1. Performance Evaluation with NSL-KDD 20% according to Attack Types. This section evaluates the DOS, Probe, U2R, and R2L, the types of attack of the NSL-KDD 20% dataset. The NID-Shield NIDS is assessed with J48 as an attribute selection approach, and finally, the selected attributes are appraised with a machine learning algorithm as a classifier.

5.1.1. Evaluation of DoS Attack with Normal and Attack Instances on Hybrid NID-Shield NIDS

(1) DoS Attack Evaluated with Hybrid NID-Shield NIDS. The following algorithms were applied for evaluation of feature subsets: attribute evaluator: CAPPER, attribute evaluator algorithm: J48, search method: best first, classifier evaluator: random forest.

The CAPPER evaluated subsets are as follows: 3, 4, 5, 6, 7, 8, 10, 12, 23, 24, 25, 29, 30, 36, 38, and 41.

In this section, the DoS attack is evaluated by the hybrid NID-Shield NIDS on the DoS attack dataset. The CAPPER

Table 11: U2R attack evaluated with hybrid NID-Shield NIDS approach.

(a)

Total instances	13,460
Correctly classified instances	13,460
Incorrectly classified instances	0
Execution time	1.86 seconds
Kappa measures	1
MAE	0.0003
RMSE	0.0066
RAE	11.2411%
RRSE	19.9452%

(b)

Accuracy	TP rate	FP rate	Precision	Recall	F-measure	MCC	ROC area	PRC area	Class
100%	1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	normal
100%	1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	buffer_overflow
100%	1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	loadmodule
100%	1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	rootkit
Weighted Avg. 100%	1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	

(c)

Confusion matrix			
a	*b*	*c*	*d*
13423	0	0	0
0	7	0	0
0	0	13	0
0	0	0	17

a—classified as normal, *b*—classified as buffer_overflow, *c*—classified as loadmodule, and *d*—classified as rootkit.

feature selector obtains the highest merit and accurate feature subsets from the combination of CFS and Wrapper approaches. Table 8 depicts the metrics of the DoS attack with its attack names classified individually. In the DoS, there are six attacks, namely, neptune, back, land, smurf, pod, and teardrop, and the normal instances. Figure 3 shows that the NID-Shield NIDS achieved an accuracy of 100% on the normal instances and 100% accuracy on the attack names such as land, back, teardrop, and neptune, while on the names of the attack such as pod and smurf, the NID-Shield NIDS achieves an accuracy of 94.7% and 99.6%, respectively. Overall, the weighted average of the accuracy of the normal and attack names is calculated; the NID-Shield NIDS achieves 100% accuracy on normal and all the attack types. Figure 4 shows the NID-Shield NIDS achieved a TP rate of 1.000 on the normal instances and a TP rate of 1.000 on attack names such as land, back, teardrop, and neptune, while on the attack names such as pod and smurf, the NID-Shield NIDS achieves a TP rate of 0.947 and 0.996, respectively. Overall, the weighted average of the TP rate is measured on normal and all attack names; the NID-Shield NIDS achieves 100% TP rate on normal and all attack names. Figure 5 depicts the FP rate evaluated by the NID-Shield NIDS on normal and attacks names, the NID-Shield NIDS achieves a 0.000 false-positive rate on all attack names, and an FP rate of 0.000 is achieved on the normal instance.

Figure 6 illustrates the precision of the NID-Shield NIDS which is assessed with normal and attack names. The NID-Shield NIDS obtained a precision of 1.000 on all normal instances and a precision of 1.000 on attack names such as neptune, back, land, and teardrop, while the precision of 0.998 and 0.973 is obtained on smurf and pod attack by the NID-Shield NIDS. Overall, a weighted average of 1.000 is obtained on precision for normal instances and attack names. Figure 7 depicts the recall appraised with NID-Shield NIDS on normal and attack names, the normal instances achieve a recall of 1.000 by the NID-Shield NIDS, and the attack names such as neptune, land, back, and teardrop achieve a recall of 1.000 by the NID-Shield NIDS, while the NID-Shield NIDS achieves a recall 0.996 and 0.947 on the attack names such as smurf and pod, respectively. Overall, the weighted average of recall is appraised for normal and all types of attack names; the NID-Shield NIDS obtained a recall of 1.000 on normal and attack names. Figure 8 depicts the F-measure of the NID-Shield NIDS appraised with the normal and attack names; the NID-Shield NIDS achieves an F

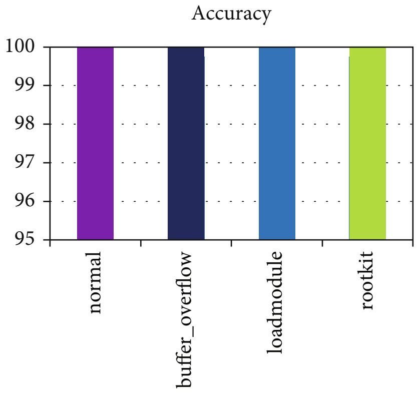


Figure 15: Accuracy of the normal and attack types evaluated by the NID-Shield NIDS on U2R attack.

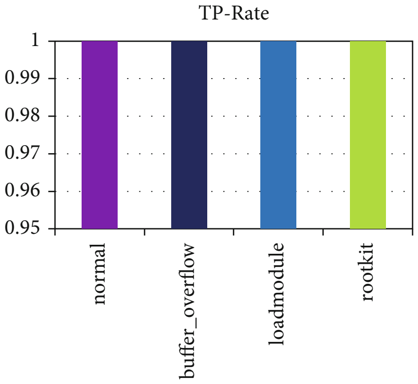


Figure 16: TP-Rate of the normal and attack types evaluated by the NID-Shield NIDS on U2R attack.

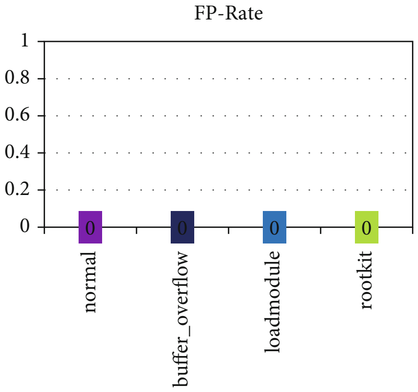


Figure 17: FP-rate of the normal and attack types evaluated by the NID-Shield NIDS on U2R attack.

-measure of 1.000 on normal instances and attack names such as neptune, land, back, and teardrop; and the NID-Shield NIDS achieves an F-measure of 1.000, while for attack names such as smurf and pod, the NID-Shield NIDS obtains the F-measure of 0.997 and 0.960, respectively. Overall, the weighted average is appraised for F-measure on normal and all attack names; the NID-Shield NIDS obtained an F -measure of 1.000 on normal and attack names. For the MCC, the NID-Shield NIDS is appraised with the normal and attack names; the NID-Shield NIDS achieves an MCC of 1.000 on normal instances and with attack names such as neptune, back, land, teardrop; and the NID-Shield NIDS achieves an MCC of 1, while for attack names such as smurf and pod, the NID-Shield NIDS obtains the MCC of 0.997 and 0.960, respectively. Overall, the weighted average of MCC is measured for normal and on all attack names; the NID-Shield NIDS obtained an MCC of 1.00, respectively.

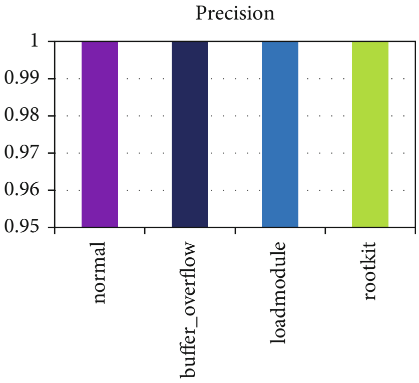


Figure 18: Precision of the normal and attack types evaluated by the NID-Shield NIDS on U2R attack.

For the ROC area, the NID-Shield NIDS achieves an overall 1.000 on normal and all attack names, respectively. For the PRC area, the NID-Shield NIDS obtained a 1.000 on normal instances, while for attack names such as land, back, teardrop, smurf, and neptune, the NID-Shield NIDS obtained a PRC area of 1.000 and for attack names called pod, the NID-Shield NIDS obtained a PRC area of 0.997. Overall, the weighted average is calculated for the PRC area on normal and all attack names; the NID-Shield NIDS achieved a PRC area of 1.000 on normal and all attack names.

5.1.2. Evaluation of Probe Attack with Normal and Attack Instances on Hybrid NID-Shield NIDS

(1) Probe Attack Evaluated with Hybrid NID-Shield NIDS. The following algorithms were applied for evaluation of feature subsets: attribute evaluator: CAPPER, attribute evaluator algorithm: J48, search method: best first, classifier evaluator: random forest.

The CAPPER evaluated subsets are as follows: 2, 3, 4, 12, 24, 27, 29, 31, 32, 35, 36, 37, and 40.

In this section, the Probe attack is evaluated with the hybrid NID-Shield NIDS on the Probe attack dataset. The stacking is applied for further improvement of the metrics. The stacked ensemble applies the random forest plus the Naive Bayes as a base classifier. Table 9 shows the Probe attack evaluation metrics without stacking ensemble, and Table 10 shows the evaluation of the Probe attack with a stacked ensemble. A considerable improvement in the FP rate is noticed when the NID-Shield NIDS is evaluated with a stacked ensemble. In the Probe attack, there are four attacks, namely, portsweep, satan, ipsweep, and nmap, and the normal instances. Figure 9 shows that the NID-Shield NIDS achieved an accuracy of 99.90% on the normal instances and for the attack names such as portsweep, satan, ipsweep, and nmap, the NID-Shield NIDS achieved an accuracy of 99.7%, 97.7%, 99.3%, and 96.3%, respectively. Overall, the weighted average of accuracy is calculated on normal and attack names; the NID-Shield NIDS obtains 99.7% accuracy on normal and on all attack names.

Figure 10 depicts that the NID-Shield NIDS achieved a TP rate of 0.999 on the normal instances and attack names such as portsweep, satan, ipsweep, and nmap; the NID-Shield NIDS achieved an accuracy of 0.997, 0.977, 0.993, and 0.963, respectively. Overall, the weighted average of the TP rate is measured on normal and attack names; the NID-Shield NIDS achieves a TP rate of 0.997 on normal and all attack names. Figure 11 depicts the FP rate evaluated by the NID-Shield NIDS on normal and attacks names; the NID-Shield NIDS achieves a 0.000 false-positive rate on attack names such as portsweep and satan; and for other attack names like ipsweep and nmap, the NID-Shield NIDS obtains an FP rate of 0.001, respectively. For the normal instance, an FPR of 0.010 is achieved by the proposed NIDS. Figure 12 depicts that the precision of the NID-Shield NIDS is assessed with normal and attack names. The NID-Shield NIDS achieves a precision of 0.998 on normal instances, and for attack names such as portsweep, satan, ipsweep, and nmap, the NID-Shield NIDS achieved a precision of 1.000, 0.993, 0.987, and 0.967, respectively.

Figure 13 depicts the recall appraised with NID-Shield NIDS on normal and attack names; the normal instances achieve a recall of 0.999, while for the attack names such as portsweep, satan, ipsweep, and nmap, the NID-Shield NIDS achieves a recall of 0.997, 0.977, 0.993, and 0.963, respectively. Overall, a weighted average of the recall is appraised for normal and on all types of attack names; the NID-Shield NIDS obtains a recall of 0.997. Figure 14 illustrates the *F*-measure of the NID-Shield NIDS assessed with the normal and attack names, the NID-Shield NIDS achieves an *F*-measure of 0.999 on normal instances, and on attack name types such as portsweep, satan, ipsweep, and nmap, the NID-Shield NIDS achieves an *F*-measure of 0.998, 0.985, 0.990, and 0.965, respectively. Overall, the weighted average is appraised for *F*-measure on normal and on all types of attack names; the NID-Shield NIDS obtained an *F* -measure of 0.997. For the MCC, the NID-Shield NIDS is appraised with the normal and attack names, the NID-Shield NIDS achieves an MCC of 0.990 on normal instances, and with attack names such as portsweep, satan, ipsweep, and nmap, the NID-Shield NIDS achieves an MCC of 0.998, 0.984, 0.990, and 0.964, respectively.

Overall, the weighted average of MCC is calculated for normal and on all attack names; the NID-Shield NIDS obtained an MCC of 0.990, respectively. The NID-Shield NIDS obtained a ROC of 0.999 on normal instances, and with attack names such as portsweep, satan, ipsweep, and nmap, the NID-Shield NIDS achieves a ROC area of 1.000, 0.995, 0.999, and 0.997, respectively. Overall, the weighted average of 0.999 is obtained by the NID-Shield NIDS in the ROC area. For the PRC area, the NID-Shield NIDS achieves 1.000 on normal instances, and with attack names such as portsweep, satan, ipsweep, and nmap, the NID-Shield NIDS achieves a PRC area of 1.000, 0.990, 0.994, and 0.986, respectively. Overall, a weighted average is appraised for the PRC area; the NID-Shield NIDS achieves a PRC area of 0.999, respectively.

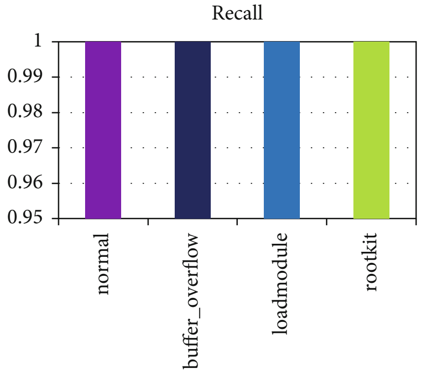


FIGURE 19: Recall of the normal and attack types evaluated by the NID-Shield NIDS on U2R attack.

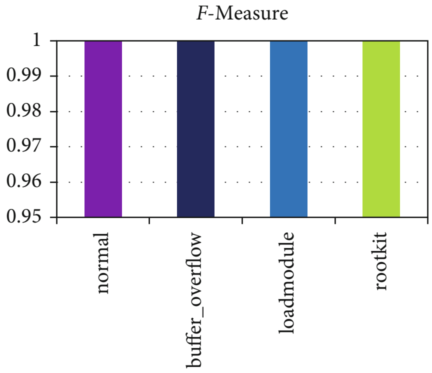


FIGURE 20: *F*-measure of the normal and attack types evaluated by the NID-Shield NIDS on U2R attack.

5.1.3. Evaluation of U2R Attack with Normal and Attack Instances on Hybrid NID-Shield NIDS

(1) U2R Attack Evaluated with Hybrid NID-Shield NIDS. The following algorithms were applied for evaluation of feature subsets: attribute evaluator: CAPPER, attribute evaluator algorithm: J48, search method: best first, classifier evaluator: random forest.

The CAPPER evaluated subsets are as follows: 3, 4, 6, 9, 10, 13, 14, 17, 18, 33, and 36.

In this section, the U2R attack is evaluated by the hybrid NID-Shield NIDS on the U2R attack dataset. Table 11 shows the metrics of the U2R attack with the three attack names in the U2R attack, namely, buffer_overflow, loadmodule, and rootkit. Figure 15 shows that the NID-Shield NIDS achieved an accuracy of 100% on the normal instances and all attack types. Figure 16 shows the NID-Shield NIDS achieved a TP rate of 1.000 on the normal instances and all attack names. Figure 17 depicts the FP rate evaluated by the NID-Shield

Table 12: R2L attack evaluated with hybrid NID-Shield NIDS approach.

(a)

Total instances	13,658
Correctly classified instances	13,648
Incorrectly classified instances	10
Execution time	1.92 seconds
Kappa measures	0.9758
MAE	0.0005
RMSE	0.0118
RAE	7.3124%
RRSE	20.4253%

(b)

Accuracy	TP rate	FP rate	Precision	Recall	*F*-measure	MCC	ROC area	PRC area	Class
100%	1.000	0.019	1.000	1.000	1.000	0.978	1.000	1.000	normal
100%	1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	ftp_write
100%	1.000	0.000	0.875	1.000	0.933	0.935	1.000	0.982	imap
100%	1.000	0.000	0.900	1.000	0.947	0.949	1.000	1.000	phf
100%	1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	multihop
100%	1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	warezmaster
97.4%	0.974	0.000	0.974	0.974	0.974	0.974	1.000	0.999	warezclient
91.7%	0.917	0.000	1.000	0.917	0.957	0.957	1.000	0.969	spy
100%	1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	gess_passwd
Weighted Avg. 99.99%	0.999	0.019	0.999	0.999	0.999	0.978	1.000	1.000	

(c)

				Confusion matrix				
a	*b*	*c*	*d*	*e*	*f*	*g*	*h*	*i*
13444	0	0	0	0	0	0	0	0
0	5	0	0	0	0	0	0	0
0	0	7	0	0	0	0	0	0
0	0	0	9	0	0	0	0	0
0	0	0	0	6	0	0	0	0
0	0	0	0	0	12	0	0	0
0	0	0	0	0	0	150	0	0
0	0	1	0	2	0	0	11	0
0	0	0	1	0	2	4	0	4

a—classified as normal, *b*—classified as ftp_write, *c*—classified as imap, *d*—classified as phf, *e*—classified as multihop, *f*—classified as warezmaster, *g*—classified as warezclient, *h*—classified as spy, *i*—classified as guess_passwd.

NIDS on normal and attacks names; the NID-Shield NIDS achieves a 0.000 false-positive rate on all attack names and normal instances. Figure 18 depicts the precision of the NID-Shield NIDS assessed with normal and attack names. The NID-Shield NIDS achieves a precision of 1.000 on all normal instances and attack names. Figure 19 depicts the recall appraised with NID-Shield NIDS on normal and attack names the normal instances and attack names achieve a recall of 1.000. Figure 20 illustrates the *F*-measure with NID-Shield NIDS evaluated with the normal instances and attack names; the NID-Shield NIDS achieves an *F*-measure of 1.000 on normal instances and attack names.

For the MCC, the NID-Shield NIDS is appraised with the normal and attack names; the NID-Shield NIDS achieves an MCC of 1.000 on normal instances and attack names. For the ROC area and PRC area, the NID-Shield NIDS achieves an overall 1.000 on normal and all attack names, respectively.

5.1.4. Evaluation of R2L Attack with Normal and Attack Instances on Hybrid NID-Shield NIDS

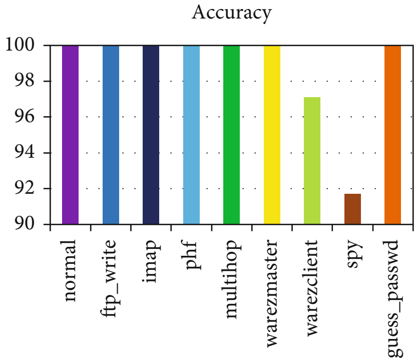


FIGURE 21: Accuracy of the normal and attack types evaluated by the NID-Shield NIDS on R2L attack.

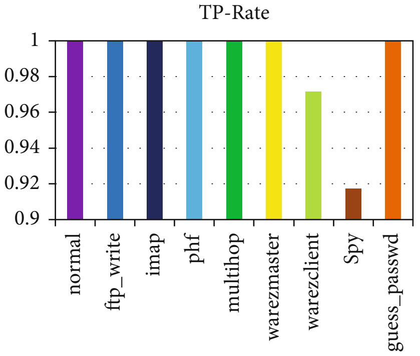


FIGURE 22: TP-rate of the normal and attack types evaluated by the NID-Shield NIDS on R2L attack.

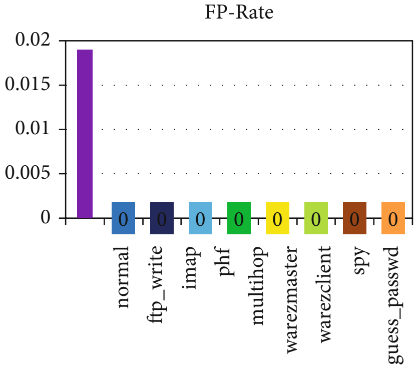


FIGURE 23: FP-rate of the normal and attack types evaluated by the NID-Shield NIDS on R2L attack.

(1) R2L Attack Evaluated with Hybrid NID-Shield NIDS. The following algorithms were applied for evaluation of feature subsets: attribute evaluator: CAPPER, attribute evaluator algorithm: J48, search method: best first, classifier evaluator: random forest.

The CAPPER evaluated subsets are as follows: 4, 5, 6, 10, 11, 17, 22, 31, 32, 33, 36, and 38.

In this section, the R2L attack is evaluated by the hybrid NID-Shield NIDS approach on the R2L attack dataset. Table 12 shows the evaluation metrics of the R2L attack. In the R2L attack, there are eight attack names, namely, ftp_write, guess_passwd, phf, imap, warezmaster, multihop, warezclient, and spy, and normal instance. Figure 21 shows that the NID-Shield NIDS achieved an accuracy of 100% on the normal instances and for attack names such as ftp_write, guess_passwd, phf, imap, warezmaster, and multihop, the NID-Shield NIDS achieved an accuracy of 100%, respectively, while for the attack names such as warezclient and spy, the NID-Shield NIDS achieves an accuracy of 97.4% and 91.7%, respectively. Overall, the weighted average in terms of accuracy is appraised for the normal and attack names; the NID-Shield NIDS achieves 99.99% accuracy on normal and all attack names. Figure 22 depicts that the NID-Shield NIDS achieved a TP rate of 1.000 on the normal instances and for the attack names such as ftp_write, guess_passwd, phf, imap, warezmaster, and multihop, the NID-Shield NIDS achieved a TP rate of 1.000, respectively, while the attack names such as warezclient and spy, the NID-Shield NIDS achieved a TP rate of 0.974 and 0.917, respectively. Overall, the weighted average of the TP rate is measured on normal and an attack name; the NID-Shield NIDS achieves a TP rate of 0.999 on normal and all attack names. Figure 23 depicts the FP rate evaluated by the NID-Shield NIDS on normal and attacks names; the NID-Shield NIDS achieves a 0.000 false-positive rate on all attack names. For the normal instance, an FPR of 0.019 is achieved. Overall, a weighted average FP rate of 0.019 is obtained on normal and attack names. Figure 24 shows that the precision of the NID-Shield NIDS is evaluated with normal and attack names. The NID-Shield NIDS achieved a precision of 1.000 on normal instances, and for attack names such as guess_passwd, ftp_write, multihop, warezmaster, and spy, the NID-Shield NIDS achieved a precision of 1.000, respectively, while for attack names such as imap, phf, and warezclient, the NID-Shield NIDS obtained a precision of 0.875, 0.900, and 0.974, respectively. Overall, a weighted average precision of 0.999 is achieved on normal and attack names. Figure 25 depicts the recall appraised with NID-Shield NIDS on normal and attack names, the normal instances achieve a recall of 1.000, and for the attack names such as guess_passwd, ftp_write, imap, phf, multihop, and warezmaster, the NID-Shield NIDS achieves a recall of 1.000, respectively, while for attack names such as warezclient and spy, a recall of

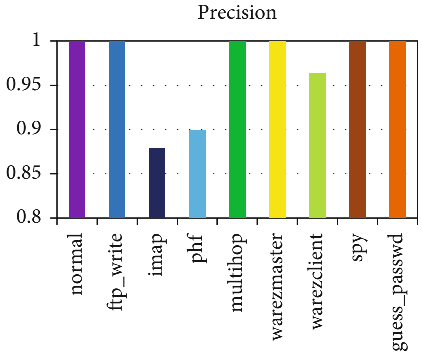


FIGURE 24: Precision of the normal and attack types evaluated by the NID-Shield NIDS on R2L attack.

0.974 and 0.917 is achieved by the proposed NIDS. Overall, the weighted average of the recall is appraised for normal and all types of attack names; the NID-Shield NIDS obtained a recall of 0.999, respectively. Figure 26 depicts the F-measure with the NID-Shield NIDS assessed with the normal and attack names, the NID-Shield NIDS achieves an F-measure of 1.000 on normal instances, and with attack names such as guess_passwd, ftp_write, multihop, and warezmaster, the NID-Shield NIDS achieves an F-measure of 1.000, respectively, while for attack names such as warezclient, spy, phf, and imap, the NID-Shield NIDS achieves an F-measure of 0.974, 0.957, 0.947, and 0.933, respectively.

Overall, the weighted average is calculated for F-measure on normal and all types of attack names; the NID-Shield NIDS obtained an F-measure of 0.999, respectively, on normal and attack names. For the MCC, the NID-Shield NIDS is appraised with the normal and attack names, the NID-Shield NIDS achieves an MCC of 0.978 on normal instances, and with attack names such as guess_passwd, ftp_write, multihop, and warezmaster, the NID-Shield NIDS achieves an MCC of 1.000, respectively, and on attack names such as imap, phf, warezclient, and spy, the NID-Shield NIDS obtained an MCC of 0.935, 0.949, 0.974, and 0.957, respectively. Overall, a weighted average is appraised for MCC; the NID-Shield NIDS achieves an MCC of 0.978 for normal and attacks names. For the ROC area, the NID-Shield NIDS achieves a 1.000 on normal instances and attack names. For the PRC area, the proposed NID-Shield NIDS achieves a 1.000 on normal instances, and with attack instances such as guess_passwd, ftp_write, phf, multihop, and warezmaster, the NID-Shield NIDS achieves a PRC area of 1.000, and for attack names such as warezclient, imap, and spy, the PRC area obtained is 0.999, 0.982, and 0.969, respectively. Overall, a weighted average PRC area of 1.000 is obtained by the NID-Shield NIDS for all normal instances and attack names.

5.1.5. Evaluation of UNSW-NB15 Dataset with Normal and Attack Instances on Hybrid NID-Shield NIDS. The following algorithms were applied for the evaluation of feature subsets: attribute evaluator: CAPPER, attribute evaluator algorithm: J48, search method: best first, classifier evaluator: random forest.

The CAPPER evaluated subsets for Reconnaissance attack are as follows: 2, 3, 7, 12, 27, 31, 36, 40, and 41.

The CAPPER evaluated subsets for Backdoor attack are as follows: 2, 3, 7, 10, 16, 27, 28, 39, and 40.

The CAPPER evaluated subsets for DoS attack are as follows: 3, 7, 8, 9, 10, 16, 31, 36, 40, and 41.

The CAPPER evaluated subsets for Exploits attack are as follows: 2, 3, 7, 8, 9, 10, 15, 17, 31, 36, and 40.

The CAPPER evaluated subsets for Analysis attack are as follows: 2, 3, 7, 8, 10, 12, 17, 28, 36, 40, and 41.

The CAPPER evaluated subsets for Fuzzers attack are as follows: 3, 7, 8, 9, 10, 12, 17, 32, and 33.

The CAPPER evaluated subsets for Worms attack are as follows: 3, 7, 8, 9, 10, 12, 17, 32, and 33.

The CAPPER evaluated subsets for Shellcode attack are as follows: 2, 3, 7, 8, 12, 27, 33, and 36.

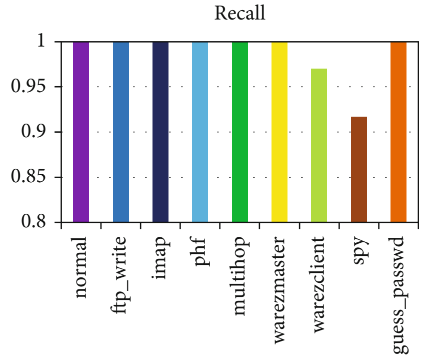


Figure 25: Recall of the normal and attack types evaluated by the NID-Shield NIDS on R2L attack.

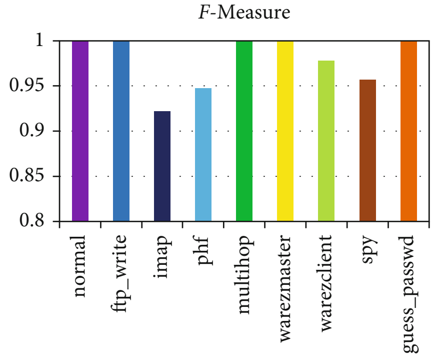


Figure 26: F-measure of the normal and attack types evaluated by the NID-Shield NIDS on R2L attack.

The CAPPER evaluated subsets for Generic attack are as follows: 2, 3, 7, 8, 9, 25, 31, 39, and 40.

In this section, the UNSW-NB15 dataset attack is evaluated by the hybrid NID-Shield NIDS approach on the UNSW-NB15 testing dataset. Table 13 illustrates the evaluation metrics of the UNSW-NB15 normal and attack instances. In the UNSW-NB15 dataset attack, there are nine attack names, namely, Backdoor, Reconnaissance, Exploits, DoS, Fuzzers, Analysis, Worms, Generic, and Shellcode, and normal instances. Figure 27 shows that the NID-Shield NIDS achieved an accuracy of 100% on the normal instances and Worms attack while for other attacks such as Backdoor, Reconnaissance, Exploits, DoS, Fuzzers, Analysis, Generic, and Shellcode, the NID-Shield NIDS achieved an accuracy of 99.71%, 99.45%, 98.70%, 99.10%, 90.14%, 99.20%, 99.70%, and 99.61%, respectively. Overall, the weighted average in terms of accuracy is appraised for the normal and an attack name; the NID-Shield NIDS achieves 99.89% accuracy on normal and all attack names. Figure 28 shows that the NID-Shield NIDS achieved a TP rate of 1 on the normal instances and Worms attack while for other attacks such as Backdoor, Reconnaissance, Exploits, DoS, Fuzzers, Analysis, Generic, and Shellcode, the NID-Shield NIDS achieved a TP rate of 0.997, 0.994, 0.987, 0.991, 0.901, 0.992, 0.997, and 0.996, respectively. Overall, the weighted average in terms of TP rate is appraised for the normal and attack names; the NID-Shield NIDS achieved an accuracy of 0.998

Table 13: UNSW-NB15 dataset evaluated with hybrid NID-Shield NIDS approach.

(a)

Total instances	1,75,341
Correctly classified instances	1, 75,183 (99.91%)
Incorrectly classified instances	158
Execution time	318.15 seconds
Kappa measures	0.9835
MAE	0.0007
RMSE	0.0121
RAE	6.3124%
RRSE	18.4253%

(b)

Accuracy	TP rate	FP rate	Precision	Recall	*F*-measure	MCC	ROC area	PRC area	Class
100%	1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	Normal
99.45%	0.994	0.007	0.996	0.998	0.997	0.995	0.999	0.999	Reconnaissance
99.71%	0.997	0.006	0.998	0.999	0.999	0.999	1.000	1.000	Backdoor
99.10%	0.991	0.007	0.995	0.991	0.997	0.997	1.000	1.000	DoS
98.70%	0.987	0.008	0.993	0.982	0.982	0.993	0.994	0.993	Exploits
99.20%	0.992	0.007	0.989	0.993	0.996	0.998	1.000	1.000	Analysis
90.14%	0.901	0.012	0.917	0.941	0.962	0.972	0.971	0.978	Fuzzers
100%	1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	Worms
99.61%	0.996	0.006	0.997	0.999	0.997	0.997	1.000	1.000	Shellcode
99.70%	0.997	0.004	0.998	0.998	0.999	0.997	1.000	1.000	Generic
Weighted Avg. 99.89%	0.998	0.006	0.999	0.998	0.997	0.992	1.000	1.000	

(c)

Confusion matrix									
a	*b*	*c*	*d*	*e*	*f*	*g*	*h*	*i*	*j*
56000	0	0	0	0	0	0	0	0	0
0	10488	0	0	0	0	0	0	0	0
0	0	1740	0	0	0	0	0	0	2
0	0	0	12260	0	0	0	0	0	0
0	2	0	0	33383	0	0	0	0	3
0	0	0	0	0	1993	0	0	0	0
0	0	3	3	7	0	18177	0	4	8
0	0	0	0	0	3	0	130	0	0
0	1	0	0	3	0	7	0	1129	0
0	0	3	1	0	4	0	0	0	39987

a—classified as Normal, *b*—classified as Reconnaissance, *c*—classified as Backdoor, *d*—classified as DoS, *e*—classified as Exploits, *f*—classified as Analysis, *g*—classified as Fuzzers, *h*—classified as Worms, *i*—classified as Shellcode, and *j*—classified as Generic.

on normal and all attack names. Figure 29 shows that the NID-Shield NIDS achieved an FP rate of 0.000 on the normal instances and Worms attack while for other attacks such as Backdoor, Reconnaissance, Exploits, DoS, Fuzzers, Analysis, Generic, and Shellcode, the NID-Shield NIDS achieved an FP rate of 0.006, 0.007, 0.008, 0.007, 0.012, 0.007, 0.004, and 0.006, respectively. Overall, the weighted average in terms of FP rate is appraised for the normal and attack names; the NID-Shield NIDS achieved an FP rate of 0.006 on normal and all attack names. Figure 30 shows that the precision of the NID-Shield NIDS is evaluated with normal instances and attack names. The NID-Shield NIDS achieved a precision of 1.000 on the normal instances and Worms attack while for other attacks such as Backdoor, Reconnaissance, Exploits, DoS, Fuzzers, Analysis, Generic, and Shellcode, the NID-Shield NIDS achieved a precision of 0.998, 0.996, 0.993, 0.995, 0.917, 0.989, 0.998, and 0.997, respectively. Overall, the weighted average in terms of precision is

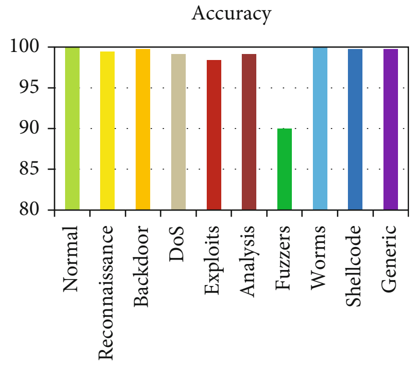


Figure 27: Accuracy of normal and attack types evaluated by the NID-Shield NIDS on UNSW-NB15 dataset.

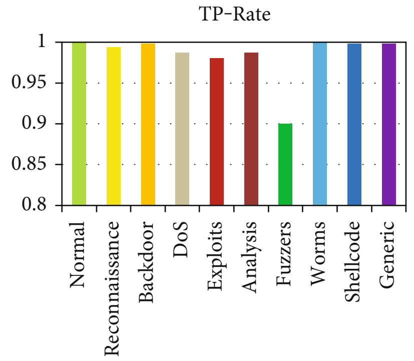


Figure 28: TP-Rate of normal and attack types evaluated by the NID-Shield NIDS on UNSW-NB15 dataset.

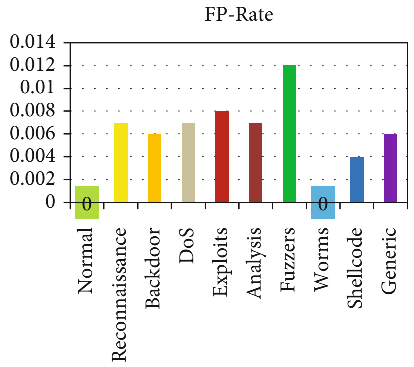


Figure 29: FP-rate of normal and attack types evaluated by the NID-Shield NIDS on UNSW-NB15 dataset.

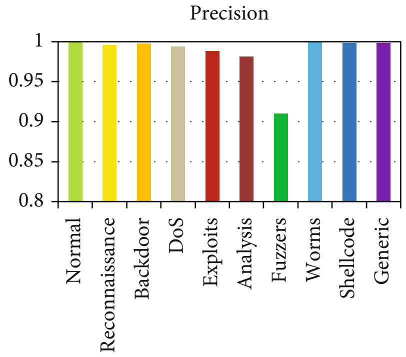


Figure 30: Precision of normal and attack types evaluated by the NID-Shield NIDS on UNSW-NB15 dataset.

appraised for the normal and attack names; the NID-Shield NIDS achieved a precision of 0.999 on normal and all attack names. Figure 31 depicts the recall appraised with NID-Shield NIDS on normal and attack names, the NID-Shield NIDS achieved a recall of 1.000 on the normal instances and Worms attack, while for other attacks such as Backdoor, Reconnaissance, Exploits, DoS, Fuzzers, Analysis, Generic, and Shellcode, the NID-Shield NIDS achieved a recall of 0.999, 0.998, 0.982, 0.991, 0.941, 0.993, 0.998, and 0.999, respectively. Overall, the weighted average in terms of recall is appraised for the normal and attack names; the NID-Shield NIDS achieved a recall of 0.998 on normal and all attack names. Figure 32 shows the *F*-measure of the NID-Shield NIDS evaluated with the normal and attack names; the NID-Shield NIDS achieved an *F*-measure of 1.000 on the normal instances and Worms attack, while for other attacks such as Backdoor, Reconnaissance, Exploits, DoS, Fuzzers, Analysis, Generic, and Shellcode, the NID-Shield NIDS achieved an *F*-measure of 0.999, 0.997, 0.982, 0.997, 0.962, 0.996, 0.999, and 0.997, respectively. Overall, the weighted average in terms of *F*-measure is appraised for the normal and attack names; the NID-Shield NIDS achieved an *F*-measure of 0.997 on normal and all attack names.

For the MCC, the NID-Shield NIDS is appraised with the normal and attack names; the NID-Shield NIDS achieved an MCC of 1.000 on the normal instances and Worms attack, while for other attacks such as Backdoor, Reconnaissance, Exploits, DoS, Fuzzers, Analysis, Generic, and Shellcode, the NID-Shield NIDS achieved an MCC of 0.999, 0.995, 0.993, 0.997, 0.972, 0.998, 0.997, and 0.997, respectively. Overall, the weighted average in terms of MCC is appraised for the normal and attack names; the NID-Shield NIDS achieved an MCC of 0.992 on normal and all attack names. The NID-Shield NIDS achieves a ROC and PRC area of 1.000 on normal and attack instances.

Table 14 shows the hybrid NID-Shield NIDS with the existing approaches in this literature. The details of the existing approaches are shown in Table 1. For the evaluation of the hybrid NID-Shield NIDS approach, the proposed hybrid NID-Shield NIDS evaluates the attack names on the UNSW-NB15 dataset, and overall performance metrics are considered such as Probe, DoS, R2L, and U2R, and attack names on the NSL-KDD 20% dataset. The NID-Shield NIDS achieves a 99.89% on the UNSW-NB15 dataset and overall accuracy of 99.90% on the NSL-KDD dataset, which is the highest among all other approaches. When the TP rate is calculated, overall, the NID-Shield NIDS obtained a TPR of 0.999 on the NSL-KDD 20% dataset and 0.9998 on the UNSW-NB15 dataset which is the best among all other approaches. When FPR is comprehensively evaluated, the literature proposed by Cavusoglu achieves an overall best FPR of 0.000035 and the NID-Shield NIDS

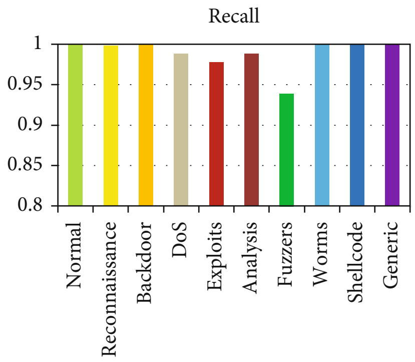


FIGURE 31: Recall of normal and attack types evaluated by the NID-Shield NIDS on UNSW-NB15 dataset.

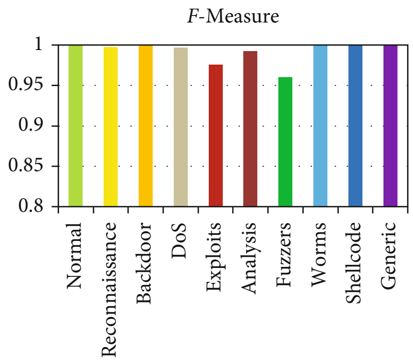


FIGURE 32: *F*-measure of normal and attack types evaluated by the NID-Shield NIDS on UNSW-NB15 dataset.

achieves a second-best FPR of 0.007 and 0.006 on NSL-KDD 20% and UNSW-NB15 datasets. The TNR is evaluated globally; the NID-Shield NIDS achieved the true negative rate of 0.993 on both datasets which are the highest among all other approaches.

The literature proposed by Arif et al. achieves the highest precision of 0.9998, and the NID-Shield NIDS achieves the second-best precision of 0.9990 on NSL-KDD 20% dataset. The NID-Shield NIDS achieves a recall of 0.999 and 0.989 on NSL-KDD 20% and UNSW-NB 15 datasets which is the highest among all other approaches. The *F*-measure is evaluated comprehensively; the NID-Shield NIDS achieves the highest *F*-measure of 0.997 and 0.999 on UNSW-NB15 and NSL-KDD 20% datasets. When MCC is appraised globally, the NID-Shield NIDS achieves the best MCC of 0.992 on both datasets which are overall best among all approaches. The ROC and PRC area of the NID-Shield NIDS is evaluated comprehensively; the NID-Shield NIDS achieves the best ROC and PRC area on both datasets which are the best among all other approaches. When the execution time is evaluated, the literature proposed by Venkataraman and Selvaraj achieves the lowest execution time of 0.23 seconds, followed by an execution time of 10.79 seconds by the literature proposed by Suad and Fadl and execution time of 10.62 seconds by the literature proposed by Cavusoglu. The NID-Shield NIDS achieves an execution time of 318.15 and 13.785 seconds on the UNSW-NB15 and NSL-KDD 20% datasets. Overall, the NID-Shield NIDS achieves the highest measures in terms of accuracy, TP rate, TNR, *F*-measure, MCC, recall, PRC, and ROC area on both the datasets.

For the insight of the discussion of the results, CAPPER and the random forest is the primary speculation for obtaining high metrics on both datasets. CAPPER is an effective feature subset selection technique that obtains accurate and high merit feature subsets from CFS and Wrapper methods. CFS searches the space of the feature subset by employing the best first search method and calculates the feature-class correlations and feature-feature correlations by applying the approaches based on conditional entropy. The high merit subset is measured by equation (3), which greatly aids in dimensionality reduction of both the testing and training data. In Wrapper, the feature subset search is executed by the best first search approach. The best first search at each iteration creates its successors having a node with maximal estimation accuracy. The induction algorithm is employed as a feature subset selection approach. The induction algorithm is run k times, and the training set uses the $k-1$ partitions, while the test set employs other partitions. Five fold cross-validation techniques are applied as the subset evaluation approach. The estimation of the accuracy is obtained by equation (4). To obtain the accurate and finest feature subsets, the machine learning approaches are applied by the Wrapper approach. The accurate and high merit feature subsets obtained by CFS and Wrapper are then combined to obtain the reduced dataset.

The random forest is considered as the most efficient classifier as compared to other classifiers. The foremost reason for obtaining the high accuracy is applying the bagging by the random forest. Employing bagging has mainly two benefits. Firstly, the accuracy is increased each time the random features are enforced. Secondly, estimation of the generalization error containing the ensemble tree combination and the correlations and its intensity appraisal is provided by the bagging. The assessment is carried out-of-bag. The main approach behind the out-of-bag estimation is the incorporation of nearly one-third of classifiers from the continuing prevailing sequence. Whenever the statistic of the sequence is incremented, the rate of error declines. Therefore, the contemporary error rate can be augmented by out-of-bag estimation; hence, it is necessary to pass on from the area where the merging of the error occurs. In the cross-validation, there is a high probability of the existence of bias; also, the degree of extent of the bias is unfamiliar, whereas the out-of-bag estimation is free from bias. The random forest applies two-thirds of the data and for testing one-third of the data from training data, to grow the tree. Out-of-bag data is simply the one-third data from the training data. Pruning is not performed by the random forest and thus aids in fast and high performance. Moreover, having the multiple-tree construction, the random forest performs reasonably well with an additional tree framework and it achieves a higher performance rather than any other decision tree method.

Table 14: Comparison of the hybrid NID-Shield NIDS with existing approaches in this study.

	Accuracy	TPR	FPR	TNR	Precision	Recall	F -measure	MCC	ROC	PRC	Time (seconds)
Proposed approach with NSL-KDD 20% dataset	99.90	0.9990	0.007	0.993	0.999	0.999	0.999	0.992	1.000	1.000	13.785
Proposed approach with UNSW-NB15 dataset	99.89	0.9989	0.006	0.993	0.999	0.989	0.997	0.992	1.000	1.000	318.15
Neha et al. [26]	99.05%	0.994	0.014	_	0.991	_	_	_	_	_	_
Arif et al. [28]	96.65%	0.9271	0.136	_	0.9998	_	_	_	_	_	_
Ahmed et al. [29]	_	0.9577	_	0.975	0.5662	_	_	_	_	_	3112.87
Tirtharaj [30]	_	0.9526	_	_	_	_	_	_	_	_	103.70
Yao et al. [31]	99.20%	0.6699	_	_	0.9655	0.967	_	_	_	_	_
Suad et al. [32]	_	_	_	_	_	_	_	_	0.995	0.962	10.79
Ijaz et al. [33]	99.8% (DoS)	_	0.17 (DoS)	_	_	_	_	_	_	_	_
Alauthaman et al. [34]	99.20%	0.9908	0.75	_	_	_	_	_	_	_	_
Venkataraman and Selvaraj [35]	83.83%	_	_	_	_	_	_	_	_	_	0.23
Kumar and Kumar [36]	99%	_	_	_	_	_	_	_	_	_	_
Cavusoglu [37]	99.86% (overall)	0.9292 (overall)	0.000035 (overall)	_	_	_	0.706 (overall)	0.954 (overall)	_	_	10.62 (overall)
Saxena et al. [38]	98.1%	0.7	_	_	_	_	_	_	_	_	_
Kambattan and Rajkumar [39]	99.45%	_	_	_	_	_	_	_	_	_	_
Kar et al. [40]	93.95%	0.955	0.1034	_	_	_	_	_	_	_	_
Mishra et al. [41]	92.12%	0.971	_	_	_	_	_	_	_	_	_
Dutta et al. [42]	91.29%	_	_	_	92.08%	90.64%	0.91	_	_	_	_
Latah and Toker [43]	84.29%	_	0.063	_	_	77.18%	84.83%	_	_	_	_
Sumaiya Thaseen et al. [44]	98.45%, on NSL-KDD dataset and 96.44% on UNSW-NB15 dataset	0.9294 on NSL-KDD dataset and 0.504 on UNSW-NB15 dataset	_	0.9438 on NSL-KDD dataset and 0.984 on UNSW-NB15 dataset	_	_	_	_	_	_	500 on NSL-KDD dataset and 1023 on UNSW-NB15 dataset
Safaldin et al. [45]	96%	0.96	0.03	_	_	_	_	_	_	_	69.6 h
Vallathan et al. [46]	98.4%	0.9602	_	0.998	_	_	_	_	_	_	_

6. Conclusion and Future Work

An efficient hybrid NID-Shield NIDS is proposed in this literature. Moreover, CAPPER an effective hybrid feature selection method is applied for accurate and highly merit feature subsets. The proposed hybrid NID-Shield NIDS classifies the UNSW-NB15 and NSL-KDD 20% dataset according to attack types and attack names. Distinct attacks may have peculiar connections as some of the attacks such as R2L and U2R may have very few N/W connections, while other attacks such as Probe and DoS may have a large number of N/W connections or can be a combination of any of them. Moreover, the hybrid NID-Shield NIDS calculates the performance metrics of attack names found in the NSL-KDD 20% dataset (DoS, Probe, U2R, and R2L) and UNSW-NB15 dataset individually. This approach further helps us to know the metrics of individual attack names and the vulnerability of the attack on the individual network. From the concluding results, it is noticed that the proposed hybrid NID-Shield NIDS with an effective CAPPER hybrid feature selection approach can improve various performance metrics on the network intrusions.

When Tables 8–14 are examined, the proposed hybrid NID-Shield NIDS obtains a comprehensive excellent performance in terms of various performance metrics on all attack types. The hybrid NID-Shield NIDS with its various parameters is investigated with existing literature studies; it has been found that the hybrid NID-Shield NIDS is the most efficient of all approaches found in the existing literature studies. In future work, we will consider applying the hybrid NID-Shield NIDS to fog computing.

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 conflict of interest.

References

[1] L. Hung-Jen and C.-h. R. Lin, "Intrusion detection system a comprehensive review," *Journal of network and applications*, vol. 36, no. 1, pp. 16–24, 2013.

[2] H. L. Motoda and H. Motoda, *Feature Selection for Knowledge Discovery and Data Mining*, vol. 454, Springer, 1998.

[3] J. P. Anderson, *Computer Security Threat Monitoring and Surveillance*, Technical report, James P. Anderson Company, Fort Washington, Pennsylvania, 1980.

[4] T. F. Lunt, J. van Horne, and L. Halme, "Automated analysis of computer system audit trails," in *Proceedings of the Ninth DOE Computer Security Group Conference*, Las Vegas, Nevada, 1986.

[5] H. S. Javitz, A. Valdes, D. E. Denning, and P. G. Neumann, *Analytical Techniques Development for a Statistical Intrusion Detection System (SIDS) Based on Accounting Records*, Technical report, SRI International, Menlo Park, California, 1986.

[6] D. Anderson, T. Frivold, and A. Valdes, *Next-Generation Intrusion Detection Expert System (NIDES). A Summary*, SRI International Computer Science Laboratory Technical Report SRI-CSL-95-07, 1995.

[7] L. D. S. Silva, A. C. Santos, T. D. Mancilha, J. D. Silva, and A. Montes, "Detecting attack signatures in the real network traffic with ANNIDA," *Expert Systems with Applications*, vol. 34, no. 4, pp. 2326–2333, 2008.

[8] A. Patcha and J. M. Park, "An overview of anomaly detection techniques: existing solutions and latest technological trends," *Computer Networks*, vol. 51, no. 12, pp. 3448–3470, 2007.

[9] C. Manikopoulos and S. Papavassiliou, "Network intrusion and fault detection. A statistical anomaly approach," *IEEE Communications Magazine*, vol. 40, no. 10, pp. 76–82, 2002.

[10] P. Fournier-Viger, C. W. Lin, A. Gomariz et al., "The SPMF open-source data mining library version 2," in *Joint European conference on machine learning and knowledge discovery in databases*pp. 36–40, Cham, Riva del Garda, Italy, 2016.

[11] P. Fournier-Viger, J. C.-W. Lin, R. U. Kiran, Y. S. Koh, and R. Thomas, "A survey of sequential pattern mining," *Data Science and Pattern Recognition*, vol. 1, no. 1, pp. 54–77, 2017.

[12] A. Smola and S. V. N. Vishwanathan, *Introduction to Machine Learning*, Cambridge University Press, 2008, ISBN-10: 0521825830.

[13] Z. Xiaojin, *Semi-Supervised Learning Literature Survey*, vol. 2, Computer Science, University of Wisconsin, Madison, 2008.

[14] S. Mukkamala, A. H. Sung, and A. Abraham, "Modeling intrusion detection systems using linear genetic programming approach," in *The 17th international conference on industrial & engineering applications of artificial intelligence and expert systems, innovations in applied artificial intelligence*, pp. 633–642, Berlin, Heidelberg, 2004.

[15] J. Pearl, "Bayesian networks. A model of self-activated memory for evidential reasoning," in *Proceedings of the 7th Conference of the Cognitive Science Society, University of California*, pp. 329–334, Irvine, CA, 2009.

[16] N. S. Altman, "An introduction to kernel and nearest-neighbor nonparametric regression (PDF)," *The American Statistician*, vol. 46, no. 3, pp. 175–185, 1992.

[17] J. B. MacQueen, "Some methods for classification and analysis of multivariate observations," in *5th Berkeley Symposium on Mathematical Statistics and Probability*, pp. 281–297, University of California Press, 1967.

[18] L. E. Baum and T. Petrie, "Statistical inference for probabilistic functions of finite state Markov chains," *The annals of mathematical statistics*, vol. 37, no. 6, pp. 1554–1563, 1966.

[19] T. Kohonen, "The self-organizing map," *Proceedings of IEEE*, pp. 1464–1480, 1990.

[20] M. Mohammed, M. B. Khan, and E. B. Bashier, *Machine Learning Algorithms and Applications*, CRC press Taylor and Francis Group, 2016, ISBN-10: 1498705383.

[21] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, vol. 2, Springer, 2009, ISBN 978-0-387-84858-7.

[22] M. Dash and H. Liu, "Feature selection for classification," *Intelligent data analysis*, vol. 1, no. 1-4, pp. 131–156, 1997.

[23] H. Liu and L. Yu, "Towards integrating feature selection algorithms for classification and clustering," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 4, pp. 491–502, 2005.

[24] R. Heady, G. Luger, A. Maccabe, and M. Servilla, *The Architecture of Network Level Intrusion Detection System*, Technical report CS90-20, Department of computer science, University of New Mexico, 1990.

[25] E. Carter, *CCSP Self-Study: Cisco Secure Intrusion Detection System (CSIDS)*, Cisco Press, 2nd edition, 2004, ISBN-10: 9781587051449.

[26] A. Neha and S. Shailendra, "An IWD-based feature selection method for intrusion detection system," *Soft computing*, vol. 22, pp. 4407–4416, 2017.

[27] H. Shah-Hosseini, "Optimization with the nature-inspired intelligent water drops algorithm," in *Evolutionary Computation*, W. P. Dos Santos, Ed., pp. 298–320, I-Tech, Vienna, 2009, ISBN 978-953-307-008-7.

[28] J. Arif, F. Malik, and K. Aslam, "A hybrid technique using binary particle swarm optimization and decision tree pruning for network intrusion detection," *Cluster Computing*, vol. 21, pp. 667–680, 2017.

[29] I. Ahmed, L. Saleh, M. Fatma, and L. Talaat, "A hybrid intrusion detection system (HIDS) based on prioritized k-nearest neighbors and optimized SVM classifiers," *Artificial Intelligence Review*, vol. 51, pp. 403–443, 2017.

[30] D. Tirtharaj, "A study on intrusion detection using neural networks trained with evolutionary algorithms," *Soft Computing*, vol. 21, pp. 2687–2700, 2017.

[31] Y. Haipeng and W. Qiyi, "An intrusion detection framework based on hybrid multi-level data mining," *International Journal of Parallel Programming*, vol. 47, pp. 740–758, 2017.

[32] M. Suad and M. Fadl, "Intrusion detection model using machine learning algorithm on Big Data environment," *Journal of big data*, vol. 5, pp. 1–12, 2018.

[33] S. Ijaz, F. A. Hashmi, S. Asghar, and M. Alam, "Vector based genetic algorithm to optimize predictive analysis in network security," *Applied intelligence*, vol. 48, no. 5, pp. 1086–1096, 2018.

[34] A. Mohammad and A. Nauman, "A P2P Botnet detection scheme based on decision tree and adaptive multilayer neural networks," *Neural Computing & Applications*, vol. 29, pp. 991–1004, 2018.

[35] V. Sivakumar and S. Rajalakshmi, "Optimal and novel hybrid feature selection framework for effective data classification," in *Advances in Systems, Control and Automation*, pp. 499–514, Springer, Singapore, 2018.

[36] K. Neeraj and K. Upendra, "Knowledge computational intelligence in network intrusion detection systems," in *Knowledge Computing and Its Applications*, pp. 161–176, Springer, Singapore, 2018.

[37] C. Unal, "A new hybrid approach for intrusion detection using machine learning methods," *Applied Intelligence*, vol. 49, pp. 2735–2761, 2019.

[38] S. Akash and S. Khushboo, "Hybrid technique based on DBSCAN for selection of improved features for intrusion detection system," in *Emerging Trends in Expert Applications and Security*, pp. 365–377, Springer, Singapore, 2019.

[39] K. Rajesh and R. Manimegalai, "An effective intrusion detection system using flawless feature selection, outlier detection and classification," in *Progress in Advanced Computing and Intelligent Engineering*, pp. 203–213, Springer, Singapore, 2019.

[40] P. Kar, S. Banerjee, K. C. Mondal, G. Mahapatra, and S. Chattopadhyay, "A hybrid intrusion detection system for hierarchical filtration of anomalies," in *Information and Communication Technology for Intelligent Systems*, vol. 106, pp. 417–426, Springer, Singapore, 2019.

[41] S. Mishra, C. Mahanty, S. Dash, and B. K. Mishra, "Implementation of BFS-NB hybrid model in intrusion detection system, recent developments in machine learning and data analytics," in *Recent Developments in Machine Learning and Data Analytics*, vol. 740, pp. 167–175, Springer, Singapore, 2019.

[42] V. Dutta, M. Choras, R. Kozik, and M. Pawlicki, "Hybrid model for improving the classification effectiveness on network intrusion detection system," in *Conference on Complex, Intelligent, and Software Intensive Systems*, Cham, 2020.

[43] M. Latah and L. Toker, "An efficient flow-based multi-level hybrid intrusion detection system for software-defined networks," *CCF Transactions on Networking*, vol. 3, pp. 26–271, 2020.

[44] I. Sumaiya Thaseen, J. Saira Banu, K. Lavanya, M. Rukunuddin Ghalib, and K. Abhishek, "An integrated intrusion detection system using correlation-based attribute selection and artificial neural network," *Transactions on Emerging Telecommunications Technologies*, vol. 32, no. 2, article e4014, 2021.

[45] M. Safaldin, M. Qtair, and L. Abualigah, "Improved binary gray wolf optimizer and SVM for intrusion detection system in wireless sensor networks," *Journal of Ambient Intelligence and Humanized Computing*, vol. 12, no. 2, pp. 1559–1576, 2021.

[46] G. Vallathan, A. John, and C. Thirumalai, "Suspicious activity detection using deep learning in secure assisted living IoT environments," *The Journal of Supercomputing*, vol. 77, pp. 3242–3260, 2021.

[47] Hackerpocalypse-cybercrime report, *In Cybersecurity Ventures*, 2016.

[48] A. AlEroud, G. Karabatis, P. Sharma, and P. He, "Context and semantics for detection of cyber attacks," *International Journal of Information and Computer Security*, vol. 6, no. 1, pp. 63–92, 2014.

[49] A. AlEroud and G. Karabatis, "Toward zero-day attack identification using linear data transformation techniques," in *IEEE 7th international conference on software security and reliability (SERE'13)*, pp. 159–168, Washington, D.C., 2013.

[50] S. Axelsson, "Intrusion detection systems: a survey and taxonomy," 2000.

[51] R. M. Snort, "Lightweight intrusion detection for networks," in *Proceedings of thirteenth USENIX conference on system administration, (LISA '99)*, pp. 229–238, Seattle, Washington, USA, 1999.

[52] J. Cannady, "Artificial neural networks for misuse detection," in *National information systems security conference*, vol. 26, pp. 368–381, Arlington, Virginia, United States, 1998.

[53] R. C. Quinlan, *4.5: Programs for Machine Learning*, Morgan Kaufmann publishers Inc, San Francisco, 1993, ISBN: 978-1-55860-238-0.

[54] T. Denoeux, "A k-nearest neighbor classification rule based on Dempster-Shafer theory," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 25, no. 5, pp. 804–813, 1995.

[55] C. Cortes and V. N. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[56] E. Alpaydin, *Introduction to Machine Learning*, MIT Press, Cambridge, 2014, ISBN: 978-0-262-028189.

[57] L. Breiman, "Random forests," *Machine Learning*, vol. 45, pp. 5–32, 2001.

[58] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees*, Wadsworth & Brooks/Cole Advanced books & Software, Monterey, CA, 1984.

[59] L. Breiman, "Bagging predictors," *Machine Learning*, vol. 24, pp. 123–140, 1996.

[60] L. Breiman, *Out-of-Bag Estimation*, Technical Report, Dept. of statistics, University of California, Berkeley, 1996.

[61] D. Mladenic and M. Grobelnik, "Feature selection for unbalanced class distribution and naive bayes," in *ICML '99: Proceedings of the Sixteenth International Conference on Machine Learning*, vol. 99, pp. 258–267, Bled, Slovenia, 1999.

[62] G. H. John, R. Kohavi, and K. Pfleger, "Irrelevant features and the subset selection problem," *Machine learning proceedings*, , pp. 121–129, 1994.

[63] P. Langley and S. Sage, "Scaling to domains with irrelevant features," in *Computational Learning Theory and Natural Learning Systems*, R. Greiner, Ed., vol. 4, MIT Press, 1994.

[64] P. Domingos and M. Pazzani, "Beyond independence: conditions for the optimality of the simple Bayesian classifier," in *Machine Learning: Proceedings of the Thirteenth International Conference on Machine Learning*, pp. 105–112, San Francisco, CA, 1996.

[65] R. C. Quinlan, *4.5: Programs for Machine Learning*, Morgan Kaufmann publishers Inc, San Francisco, 1993.

[66] J. D. Rodriguez, A. Perez, and J. A. Lozano, "Sensitivity analysis of k-fold cross validation in prediction error estimation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 3, pp. 569–575, 2010.

[67] N. Thomas Rincy and R. Gupta, "An efficient feature subset selection approach for machine learning," *Multimedia tools and applications*, vol. 80, pp. 12737–12830, 2021.

[68] Z. H. Zhou, *Ensemble Methods Foundation and Algorithms*, CRC press: Taylor and Francis Group, 2012.

[69] P. Smyth and D. Wolpert, "Stacked density estimation," in *Advances in Neural Information Processing Systems*, pp. 668–674, MIT Press, Cambridge, MA, 1998.

[70] S. Samdani and S. Shukla, "A novel technique for converting nominal attributes to numeric attributes for intrusion detection," in *2017 8th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, pp. 1–5, Delhi, 2017.

[71] A. Binbusayyis and T. Vaiyapuri, "Comprehensive analysis and recommendation of feature evaluation measures for intrusion detection," *Heliyon*, vol. 6, no. 7, 2020.

[72] S. Revathi and A. Malathi, "A detailed analysis on NSL-KDD dataset using various machine learning," *International Journal of Engineering Research & Technology*, vol. 2, no. 12, pp. 1848–1853, 2013.

[73] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP-'99 data set," in *Proceedings of the IEEE Symposium on Computational Intelligence in Security and Defense Applications*, Ottawa, Canada, 2009.

[74] P. Kavitha and M. Usha, "Anomaly based intrusion detection in WLAN using discrimination algorithm combined with Naïve Bayesian classifier," *Journal of Theoretical and Applied Information Technology*, vol. 62, no. 1, pp. 77–84, 2014.

[75] N. Moustafa and J. Slay, "UNSW-NB15 a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," in *Military Communications and Information Systems Conference (MilCIS)*, pp. 1–6, Canberra, 2015.

[76] I. H. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques*, Morgan Kaufmann, San Francisco, 2nd edition, 2005.

[77] H. Hanan and B. David, "A taxonomy and survey of intrusion detection system design techniques, network threats and datasets," pp. 1–35, 2018, https://arxiv.org/abs/1806.03517.

Hindawi
Wireless Communications and Mobile Computing
Volume 2021, Article ID 2034125, 8 pages
https://doi.org/10.1155/2021/2034125

Research Article

DeepLab and Bias Field Correction Based Automatic Cone Photoreceptor Cell Identification with Adaptive Optics Scanning Laser Ophthalmoscope Images

Yiwei Chen,[1] Yi He,[1] Jing Wang,[1,2] Wanyue Li,[1,2] Lina Xing,[1] Xin Zhang,[1] and Guohua Shi[1,2,3]

[1] *Jiangsu Key Laboratory of Medical Optics, Suzhou Institute of Biomedical Engineering and Technology, Chinese Academy of Sciences, Suzhou 215163, China*
[2] *Department of Biomedical Engineering, University of Science and Technology of China, Hefei 230041, China*
[3] *Center for Excellence in Brain Science and Intelligence Technology, Chinese Academy of Sciences, Shanghai 200031, China*

Correspondence should be addressed to Guohua Shi; ghshi_lab@126.com

Received 3 May 2021; Revised 24 May 2021; Accepted 1 June 2021; Published 12 June 2021

Academic Editor: Yulin Wang

The identification of cone photoreceptor cells is important for early diagnosing of eye diseases. We proposed automatic deep-learning cone photoreceptor cell identification on adaptive optics scanning laser ophthalmoscope images. The proposed algorithm is based on DeepLab and bias field correction. Considering manual identification as reference, our algorithm is highly effective, achieving precision, recall, and $F1$ score of 96.7%, 94.6%, and 95.7%, respectively. To illustrate the performance of our algorithm, we present identification results for images with different cone photoreceptor cell distributions. The experimental results show that our algorithm can achieve accurate photoreceptor cell identification on images of human retinas, which is comparable to manual identification.

1. Introduction

Vision is one of the most important human senses. Unfortunately, as a major cause of blindness, retinopathy has become increasingly common. Most retinopathy patients can prevent blindness with early diagnosis and treatment, which provide promising outcomes. Although optical imaging allows observing the retina, higher-resolution imaging is required for the early diagnosis of retinopathy. However, ocular aberrations limit the resolution of optical imaging. To address this limitation, adaptive optics (AO), which was originally intended for removing aberrations caused by atmospheric instability [1], has been used to correct ocular aberrations in retinal imaging [2–4]. AO allows the resolution of *in vivo* retinal imaging to reach the cellular level [4–6]. In particular, AO scanning laser ophthalmoscopy (AO-SLO) uses an integrated AO for clearly imaging cone photoreceptor cells [4]. Thus, AO-SLO allows to observe pathological changes in the distribution of photoreceptor cells on the retina, thus, outperforming other retinal imaging techniques in the diagnosis of certain diseases characterized by disorders in the distribution of cone photoreceptor cells [7–11].

To quantitatively calculate the distribution of cone photoreceptor cells, individual cells should be identified. Although manual identification of cone photoreceptor cells is reliable, it is time-consuming and subjective. Therefore, semiautomatic and automatic algorithms for cone photoreceptor-cell identification have been devised [12–26]. They can be nonlearning-based algorithms [12–18], supervised-learning algorithms [19–23], and unsupervised-learning algorithms [24–26]. Among them, supervised deep-learning algorithms have achieved the highest accuracy, thus, being a promising research direction given their potentially high performance.

In 2014, Google introduced a supervised deep-learning semantic segmentation model called DeepLab [27]. With

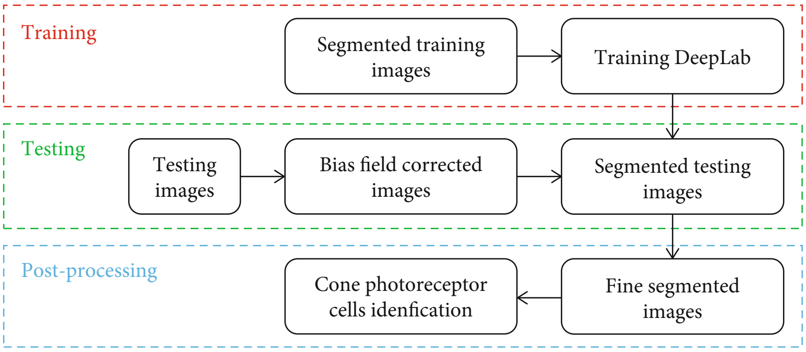


FIGURE 1: Outline of proposed AO-SLO cone photoreceptor identification algorithm based on DeepLab and bias field correction.

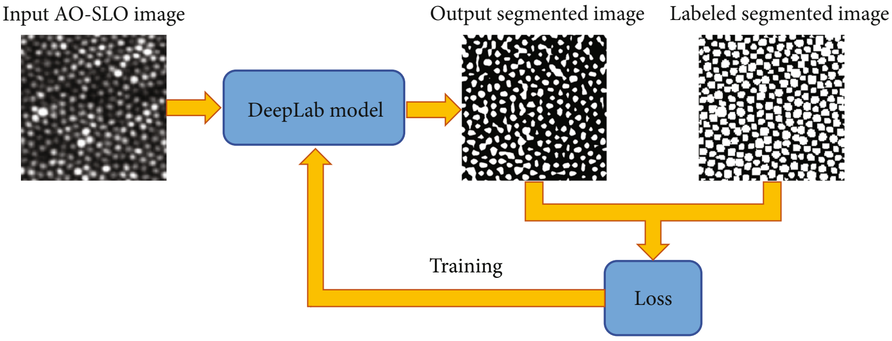
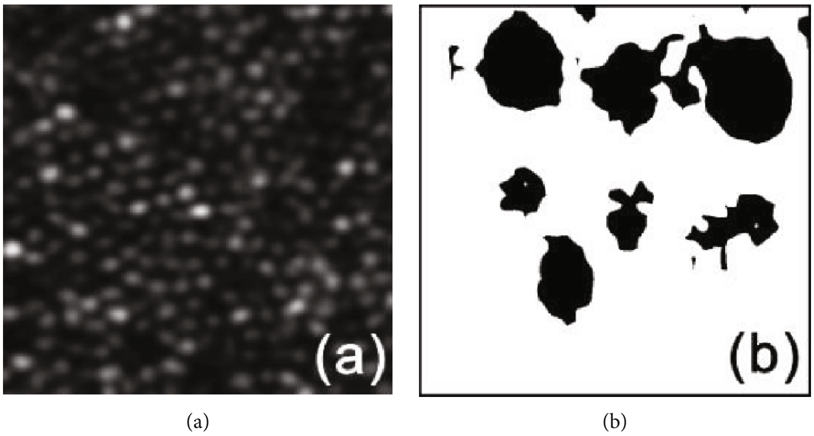


FIGURE 2: Outline of training process.



(a)

(b)

FIGURE 3: Representative example of failed segmentation by directly using AO-SLO images and trained DeepLab. (a) AO-SLO image and (b) segmentation results.

remarkable advantages, DeepLab has become a hot topic in research and engineering [28–33], and one of its popular variants, DeepLab v3 [34], has been widely used in medical image processing [35–41]. We propose an automatic cone photoreceptor cell identification algorithm based on DeepLab v3 for AO-SLO images. The proposed algorithm also uses bias field correction [42] to further improve the identification accuracy. To confirm the effectiveness of the proposed algorithm, we determined various evaluation measures (i.e., precision, recall, and $F1$ score) with respect to manual identification, which is considered as the reference providing the ground truth. The performance of the proposed algorithm is further demonstrated by showing cone photoreceptor-cell

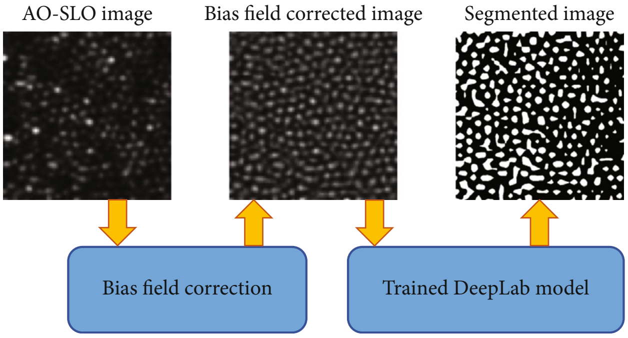
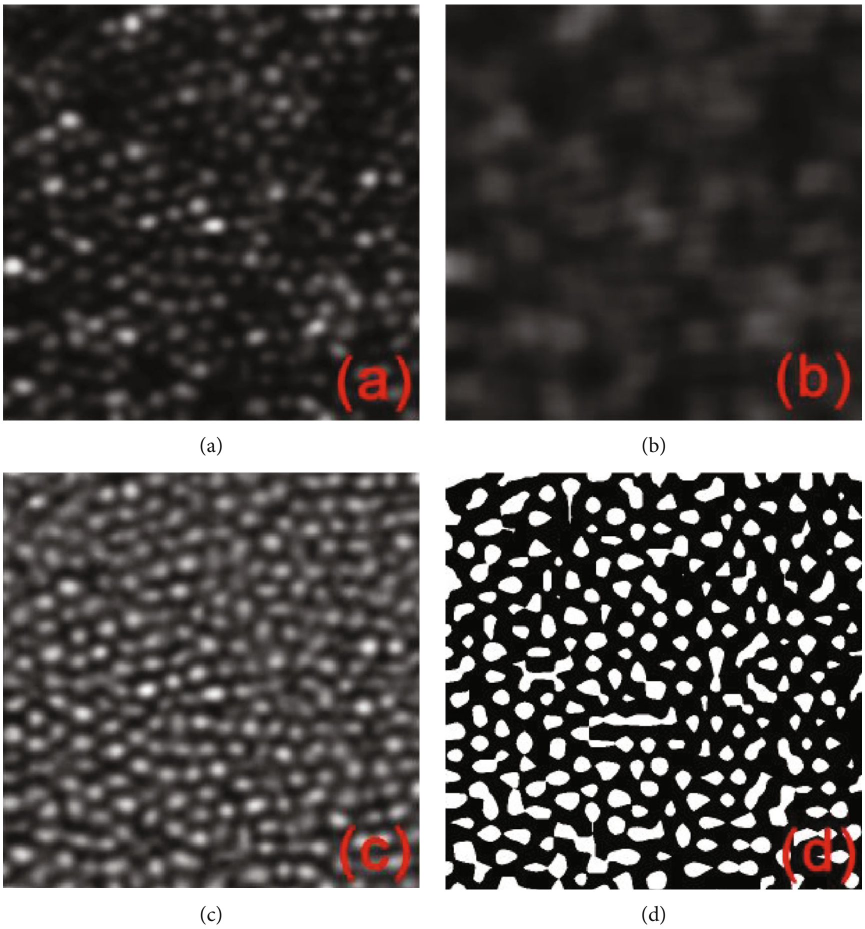


FIGURE 4: Outline of testing process.



(a)

(b)

(c)

(d)

Figure 5: Representative example of bias field correction and DeepLab segmentation. (a) AO-SLO image, (b) bias field image, (c) Bias-field-corrected image, and (d) segmentation results.

identification results for AO-SLO images with different cell distributions.

2. Methods

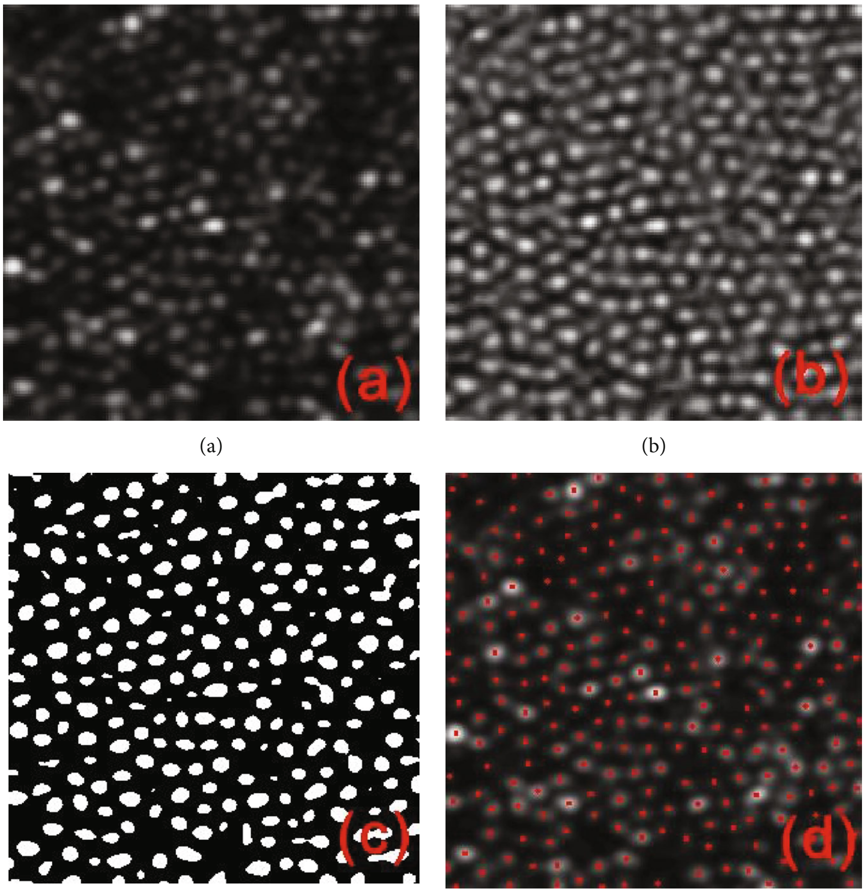
Figure 1 shows the outline of the proposed deep-learning cone photoreceptor-cell identification algorithm with its main steps of (1) training, (2) testing, and (3) postprocessing. First, the training dataset that includes AO-SLO images and their corresponding segmented images is used to train DeepLab [34]. Second, the bias-field-corrected images obtained from the test dataset after applying bias field correction [42] are input to the trained DeepLab [32] to generate segmented test images. Third, the bias-field-corrected images and segmented test images are processed by threshold-based algorithm to obtain finely segmented images to identify individual cone photoreceptor cells by calculating their centroids.

2.1. Training. To achieve a fine segmentation of cone photoreceptor cells, we magnified the training AO-SLO images and their corresponding segmented images four times isotropically before training segmentation. In detail, the training AO-SLO images were interpolated using the antialiasing mode to obtain high-quality images, and the corresponding segmented images were interpolated using the nearest mode for binarization. Both interpolation operations are available in Python Imaging Library. Then, DeepLab v3 [34] with its ResNet-101 backbone pretrained on the ImageNet dataset was trained using the magnified images. In the training images, the area of the cone photoreceptor cells is larger than that of the background. To compensate for such imbalance, we introduced a cross-entropy loss function that weights the cone photoreceptor cells (0.3) and background (0.7) separately. During training, we set the batch size and number of epochs to 2 and 100, respectively. The outline of the training process is shown in Figure 2.

2.2. Testing. The direct usage of the trained DeepLab v3 to segment four-time magnified test AO-SLO images can cause failure with high probability. A representative example of a failure case is shown in Figure 3, where segmentation is based on local intensity bias instead of cone photoreceptor cells, leading to segmentation failure.

To solve this problem, we applied bias field correction to the AO-SLO images. First, a bias field image is generated by applying a Gaussian filter whose sigma value is 22 pixels length to the AO-SLO image [26]:

$$\text{Bias field image} = \text{Gaussian filter}(\text{AO-SLO image}). \qquad (1)$$



(a)

(b)

(c)

(d)

Figure 6: Representative example of postprocessing. (a) AO-SLO image, (b) bias field corrected image, (c) thresholding results, and (d) cone photoreceptor cell identification results.

Table 1: Performance measures obtained from cone photoreceptor cell identification algorithms.

Methods	Precision	Recall	*F*1 score
Graph theory based algorithm [15]	98.2%	98.5%	98.3%
Proposed algorithm	96.7%	94.6%	95.7%
Watershed based algorithm [18]	93.2%	96.6%	94.9%
K-means clustering based algorithm [26]	93.4%	95.2%	94.3%
Superpixels based algorithm [25]	80.1%	93.5%	86.3%

Second, the AO-SLO image is corrected by extracting the bias field image [42]:

$$\text{Bias field corrected image}(x, y) = \text{Mean}(\text{Bias field image}) \times \frac{\text{AO-SLO image}(x, y)}{\text{Bias field image}(x, y)}. \quad (2)$$

Third, the four-time magnified bias-field-corrected image is input to the trained DeepLab, and the segmentation results are obtained. The outline of the testing process is shown in Figure 4.

Figure 5 depicts the bias field correction [42] and DeepLab segmentation [34] performed on the image shown in Figure 3. The bias field is corrected, and the segmented image is accurate.

2.3. Postprocessing. Figure 5 shows that some cone photoreceptor cells are merged after DeepLab segmentation. To mitigate this problem, we applied thresholding to the bias-field-corrected images [36]. The intensity values in the DeepLab segmentation mask were first extracted from the bias-field-corrected image. Then, the mean intensity value was calculated and used as the threshold to segment the bias-field-corrected image. Through thresholding, cone photoreceptor cells were identified in two steps. In detail, the contours of the segmentation results were extracted using function find Contours of OpenCV, and the centroids of the areas inside

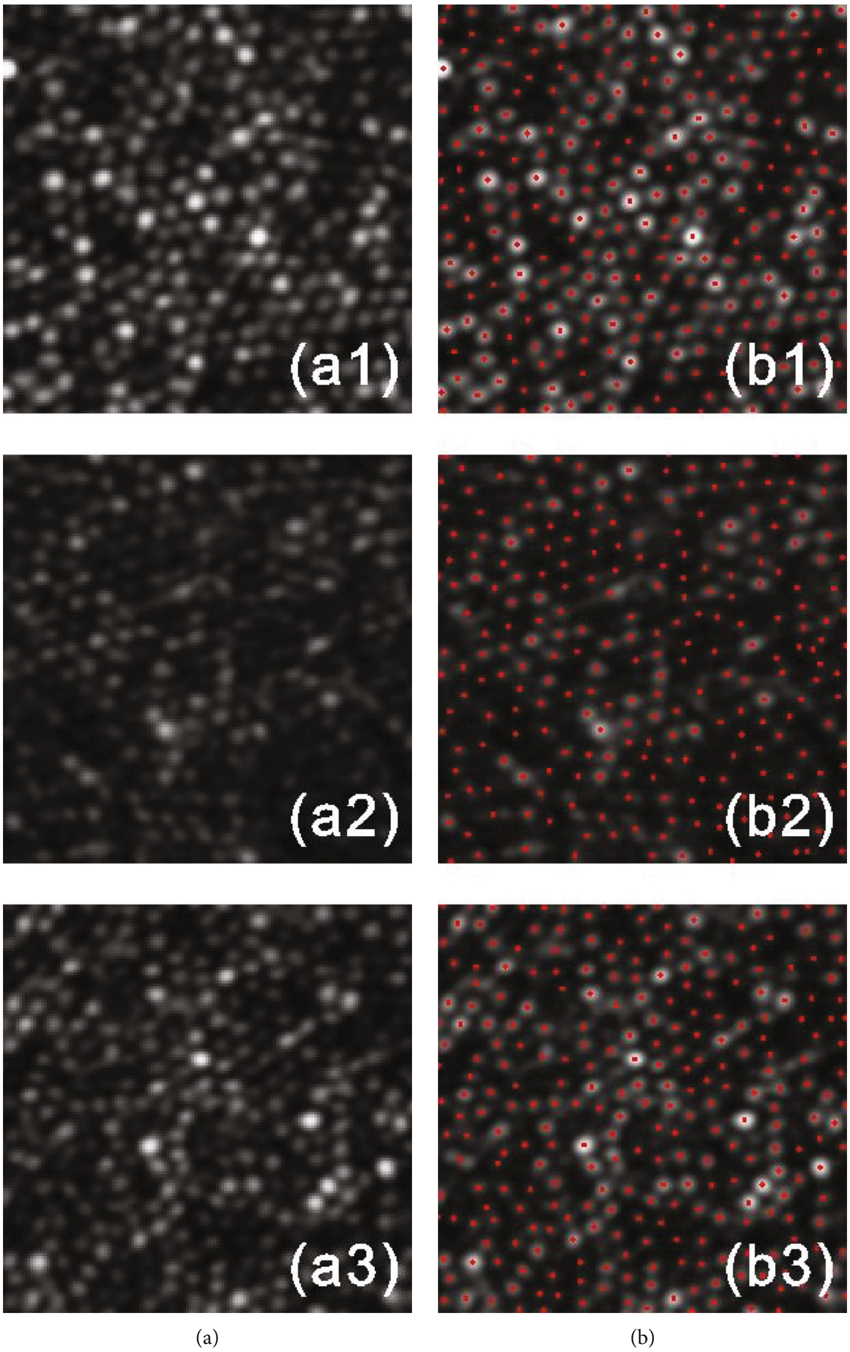


Figure 7: Results of proposed algorithm with different cone photoreceptor cell distributions. (a) Input AO-SLO images and (b) identified cone photoreceptor cells.

the contours were then considered as identified cone photoreceptor cells. A representative example of postprocessing is shown in Figure 6, where adjacent cell merging is mostly solved, and individual cone photoreceptor cells are accurately identified.

3. Results

We evaluated the proposed algorithm on a publicly available dataset [15] that contains 840 AO-SLO images and their corresponding cone photoreceptor cell segmentation results as ground truth [15]. We used 800 AO-SLO images for the training dataset, and the remaining 40 images for the test dataset. The automatic processing took 2.95 hours for training with two batch sizes over 100 epochs, 8.77 s for testing, and 0.76 s for postprocessing. These computation times were obtained on a computer running 64-bit Python and equipped with an Intel Core i7-10870H processor (2.20 GHz), 16.0 GB RAM, and NVIDIA GeForce RTX 2060 graphics card.

To confirm the effectiveness of the proposed algorithm for cone photoreceptor cell identification, we evaluated its identification performance regarding three measures, namely, precision, recall, and $F1$ score, with respect to the manual identification results taken as reference. The overall precision, recall, and $F1$ score for identification are listed in Table 1, where the values are compared with those of several algorithms [15, 18, 25, 26]. The proposed algorithm achieves accurate cone photoreceptor cell identification, outperforming the comparison algorithm [18, 25, 26] except the graph theory-based algorithm [15] which is often referred to as ground truthing cone photoreceptor cell identification but needs a large amount of computing and complex implementation.

To illustrate the performance of the proposed algorithm, Figure 7 shows cone photoreceptor cell identification results for different cone photoreceptor cell distributions on AO-SLO images. The cone photoreceptor cells are accurately identified on the three AO-SLO images with different distributions.

4. Discussion

In semantic segmentation, the relationship between the target segmentation objects and background is usually complex. Cone photoreceptor cell identification is relatively simple: (1) only one type of object, a cone photoreceptor cell, should be segmented; (2) cone photoreceptor cells do not contain rich texture details. Thus, an algorithm can segment the images according to area-based information. As the target area containing the cone photoreceptor cells is much larger than the area in general semantic segmentation, DeepLab algorithm is trained with bias if the cone photoreceptor cells and background are weighted equally. To prevent bias, we designed a cross-entropy loss function with a smaller weight given to cone photoreceptor cells.

In general, supervised deep-learning algorithms provide higher accuracy than nonlearning-based and unsupervised-learning algorithms. Therefore, automatic algorithms for the accurate identification of cone photoreceptor cells on AO-SLO images can be developed by applying and modifying deep learning algorithms, which have demonstrated high-performance image segmentation and identification but have not yet been used for cone photoreceptor cell identification. In this regard, we presented the modified versions of three famous methods [43–45] as promising solutions for developing automatic and accurate cone photoreceptor cell identification algorithms on AO-SLO images.

5. Conclusions

We propose an automatic deep-learning algorithm for the identification of cone photoreceptor cells on AO-SLO images. The algorithm implements DeepLab v3 and bias field correction as its core techniques. To confirm the effectiveness of the proposed algorithm, we obtained its precision, recall, and $F1$ score with respect to manual identification, obtaining values of 96.7%, 94.6%, and 95.7%, respectively. Furthermore, to illustrate the performance of the proposed algorithm, we obtained the cone photoreceptor cell identification results for different cone photoreceptor cell distributions on AO-SLO images.

Data Availability

The original dataset used in this paper is a publicly available dataset which can be obtained online (http://people.duke.edu/~sf59/Chiu_BOE_2013_dataset.htm) [25]. However, our source codes are not publicly available due to them containing information that could compromise research participant privacy.

Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

Acknowledgments

This work was supported in part by the Natural Science Foundation of Jiangsu Province (BK20200214), National Key R&D Program of China (2017YFB0403701), Jiangsu Province Key R&D Program (BE2019682, BE2018667), National Natural Science Foundation of China (61605210, 61675226, 61378090), Youth Innovation Promotion Association of Chinese Academy of Sciences (2019320), Frontier Science Research Project of the Chinese Academy of Sciences (QYZDB-SSW-JSC03), and Strategic Priority Research Program of the Chinese Academy of Sciences (XDB02060000).

References

[1] H. W. Babcock, "The possibility of compensating astronomical seeing," *Publications of the Astronomical Society of the Pacific*, vol. 65, no. 386, pp. 229–236, 1953.

[2] A. Roorda, F. Romero-Borja, W. J. Donnelly III, H. Queener, T. J. Hebert, and M. C. Campbell, "Adaptive optics scanning laser ophthalmoscopy," *Optics Express*, vol. 10, no. 9, pp. 405–412, 2002.

[3] S. A. Burns, R. Tumbar, A. E. Elsner, D. Ferguson, and D. X. Hammer, "Large-field-of-view, modular, stabilized, adaptive-optics-based scanning laser ophthalmoscope," *Journal of the Optical Society of America. A*, vol. 24, no. 5, pp. 1313–1326, 2007.

[4] R. D. Ferguson, Z. Zhong, D. X. Hammer et al., "Adaptive optics scanning laser ophthalmoscope with integrated wide-field retinal imaging and tracking," *Journal of the Optical Society of America. A*, vol. 27, no. 11, pp. 265–277, 2010.

[5] Y. Kitaguchi, K. Bessho, T. Yamaguchi, N. Nakazawa, T. Mihashi, and T. Fujikado, "In vivo measurements of cone photoreceptor spacing in myopic eyes from images obtained by an adaptive optics fundus camera," *Japanese Journal of Ophthalmology*, vol. 51, no. 6, pp. 456–461, 2007.

[6] A. Reumueller, "Three-dimensional composition of the photoreceptor cone layers in healthy eyes using adaptive-optics optical coherence tomography (AO-OCT)," *PLoS One*, vol. 16, no. 1, article e0245293, 2021.

[7] J. Lammer, S. G. Prager, M. C. Cheney et al., "Cone photoreceptor irregularity on adaptive optics scanning laser ophthalmoscopy correlates with severity of diabetic retinopathy and macular edema," *Investigative Ophthalmology & Visual Science*, vol. 57, no. 15, pp. 6624–6632, 2016.

[8] S. P. Park, W. Lee, E. J. Bae et al., "Early structural anomalies observed by high-resolution imaging in two related cases of

autosomal-dominant retinitis pigmentosa," *Ophthalmic Surgery, Lasers and Imaging Retina*, vol. 45, no. 5, pp. 469–473, 2014.

[9] S. Nakatake, Y. Murakami, J. Funatsu et al., "Early detection of cone photoreceptor cell loss in retinitis pigmentosa using adaptive optics scanning laser ophthalmoscopy," *Graefe's Archive for Clinical and Experimental Ophthalmology*, vol. 257, no. 6, pp. 1169–1181, 2019.

[10] R. L. Steinmetz, A. Garner, J. I. Maguire, and A. C. Bird, "Histopathology of incipient fundus flavimaculatus," *Ophthalmology*, vol. 98, no. 6, pp. 953–956, 1991.

[11] Y. Chen, K. Ratnam, S. M. Sundquist et al., "Cone photoreceptor abnormalities correlate with vision loss in patients with Stargardt disease," *Investigative Ophthalmology & Visual Science*, vol. 52, no. 6, pp. 3281–3292, 2011.

[12] A. Turpin, P. Morrow, B. Scotney, R. Anderson, and C. Wolsley, "Automated identification of photoreceptor cones using multi-scale modelling and normalized cross-correlation," in *Image Analysis and Processing – ICIAP 2011. ICIAP 2011. Lecture Notes in Computer Science, vol 6978*pp. 494–503, Springer, Berlin, Heidelberg.

[13] D. Cunefare, R. F. Cooper, B. Higgins et al., "Automatic detection of cone photoreceptors in split detector adaptive optics scanning light ophthalmoscope images," *Biomedical Optics Express*, vol. 7, no. 5, pp. 2036–2050, 2016.

[14] D. M. Bukowska, A. L. Chew, E. Huynh et al., "Semi-automated identification of cones in the human retina using circle Hough transform," *Biomedical Optics Express*, vol. 6, no. 12, pp. 4676–4693, 2015.

[15] S. J. Chiu, Y. Lokhnygina, A. M. Dubis et al., "Automatic cone photoreceptor segmentation using graph theory and dynamic programming," *Biomedical Optics Express*, vol. 4, no. 6, pp. 924–937, 2013.

[16] K. Y. Li and A. Roorda, "Automated identification of cone photoreceptors in adaptive optics retinal images," *Journal of the Optical Society of America A*, vol. 24, no. 5, pp. 1358–1363, 2007.

[17] J. Liu, H. Jung, A. Dubra, and J. Tam, "Automated photoreceptor cell identification on nonconfocal adaptive optics images using multiscale circular voting," *Investigative Ophthalmology and Visual Science*, vol. 58, no. 11, pp. 4477–4489, 2017.

[18] Y. Chen, Y. He, J. Wang et al., "Automated cone photoreceptor cell segmentation and identification in adaptive optics scanning laser ophthalmoscope images using morphological processing and watershed algorithm," *IEEE Access*, vol. 8, pp. 105786–105792, 2020.

[19] D. Cunefare, C. S. Langlo, E. J. Patterson et al., "Deep learning based detection of cone photoreceptors with multimodal adaptive optics scanning light ophthalmoscope images of achromatopsia," *Biomedical Optics Express*, vol. 9, no. 8, pp. 3740–3756, 2018.

[20] D. Cunefare, A. L. Huckenpahler, E. J. Patterson, A. Dubra, J. Carroll, and S. Farsiu, "RAC-CNN: multimodal deep learning based automatic detection and classification of rod and cone photoreceptors in adaptive optics scanning light ophthalmoscope images," *Biomedical Optics Express*, vol. 10, no. 8, pp. 3815–3832, 2019.

[21] J. Hamwood, D. Alonso-Caneiro, D. M. Sampson, M. J. Collins, and F. K. Chen, "Automatic detection of cone photoreceptors with fully convolutional networks," *Translational Vision Science & Technology*, vol. 8, no. 6, p. 10, 2019.

[22] D. Cunefare, L. Fang, R. F. Cooper, A. Dubra, J. Carroll, and S. Farsiu, "Open source software for automatic detection of cone photoreceptors in adaptive optics ophthalmoscopy using convolutional neural networks," *Scientific Reports*, vol. 7, no. 1, pp. 1–11, 2017.

[23] B. Davidson, A. Kalitzeos, J. Carroll et al., "Automatic cone photoreceptor localisation in healthy and Stargardt afflicted retinas using deep learning," *Scientific Reports*, vol. 8, no. 1, pp. 1–13, 2018.

[24] C. Bergeles, A. M. Dubis, B. Davidson et al., "Unsupervised identification of cone photoreceptors in non-confocal adaptive optics scanning light ophthalmoscope images," *Biomedical Optics Express*, vol. 8, no. 6, pp. 3081–3094, 2017.

[25] Y. Chen, Y. He, J. Wang et al., "Automated superpixels-based identification and mosaicking of cone photoreceptor cells for adaptive optics scanning laser ophthalmoscope," *Chinese Optics Letters*, vol. 18, no. 10, article 101701, 2020.

[26] Y. Chen, Y. He, J. Wang et al., "Automated cone cell identification on adaptive optics scanning laser ophthalmoscope images based on TV-L1 optical flow registration and K-means clustering," *Applied Sciences*, vol. 11, no. 5, article 2259, 2021.

[27] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Semantic image segmentation with deep convolutional nets and fully connected CRFS," 2014, http://arxiv.org/abs/1412.7062.

[28] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3431–3440, Boston, MA, USA, June 2015.

[29] W. Liu, D. Anguelov, D. Erhan et al., "SSD: single shot multibox detector," in *Computer Vision – ECCV 2016. ECCV 2016. Lecture Notes in Computer Science, vol 9905*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds., Springer, Cham, 2016.

[30] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1125–1134, Honolulu, HI, USA, July 2017.

[31] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFS," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 4, pp. 834–848, 2018.

[32] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: a deep convolutional encoder-decoder architecture for image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 12, pp. 2481–2495, 2017.

[33] M. Cordts, M. Omran, S. Ramos et al., "The cityscapes dataset for semantic urban scene understanding," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA, June 2016.

[34] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," 2017, http://arxiv.org/abs/1706.05587.

[35] W.-T. Xiao, L.-J. Chang, and W.-M. Liu, "Semantic segmentation of colorectal polyps with DeepLab and LSTM networks," in *2018 IEEE International Conference on Consumer Electronics-Taiwan (ICCE-TW)*, Taichung, Taiwan, May 2018.

[36] Y. Wang, S. Sun, J. Yu, and D. Yu, "Skin lesion segmentation using atrous convolution via DeepLab V3," 2018, http://arxiv.org/abs/1807.08891.

[37] E. Grøvik, D. Yi, M. Iv et al., “Handling missing MRI sequences in deep learning segmentation of brain metastases: a multicenter study,” *NPJ Digital Medicine*, vol. 4, no. 1, pp. 33–37, 2021.

[38] L. Ahmed, M. M. Iqbal, H. Aldabbas, S. Khalid, Y. Saleem, and S. Saeed, “Images data practices for semantic segmentation of breast cancer using deep neural network,” *Journal of Ambient Intelligence and Humanized Computing*, pp. 1–17, 2020.

[39] A. Subramanian and K. Srivatsan, *Exploring Deep Learning Based Approaches for Endoscopic Artefact Detection and Segmentation*, EndoCV@ ISBI, 2020.

[40] C.-H. Huang, W.-T. Xiao, L.-J. Chang, W.-T. Tsai, and W.-M. Liu, “Automatic tissue segmentation by deep learning: from colorectal polyps in colonoscopy to abdominal organs in CT exam,” in *2018 IEEE Visual Communications and Image Processing (VCIP)*, Taichung, Taiwan, December 2018.

[41] D. Yi, E. Gøvik, M. Iv, E. Tong, G. Zaharchuk, and D. Rubin, “Random bundle: brain metastases segmentation ensembling through annotation randomization,” 2020, http://arxiv.org/abs/2002.09809.

[42] P. Zang, G. Liu, M. Zhang et al., “Automated motion correction using parallel-strip registration for wide-field en face OCT angiogram,” *Biomedical Optics Express*, vol. 7, no. 7, pp. 2823–2836, 2016.

[43] G. Lin, A. Milan, C. Shen, and I. Reid, “Refinenet: multi-path refinement networks for high-resolution semantic segmentation,” in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, USA, July 2017.

[44] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, “Pyramid scene parsing network,” in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, USA, July 2017.

[45] C. Peng, X. Zhang, G. Yu, G. Luo, and J. Sun, “Large kernel matters — improve semantic segmentation by global convolutional network,” in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, USA, July 2017.

Hindawi
Wireless Communications and Mobile Computing
Volume 2021, Article ID 5582341, 10 pages
https://doi.org/10.1155/2021/5582341

Research Article

An Approach to Spatiotemporal Trajectory Clustering Based on Community Detection

Xin Wang,[1] **Xinzheng Niu**,[1] **Jiahui Zhu**,[1] **and Zuoyan Liu**[2]

[1] *School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, Sichuan, China*
[2] *Department of Rehabilitation Medical Center, West China Hospital, West China School of Nursing, Sichuan University, Chengdu, China*

Correspondence should be addressed to Zuoyan Liu; zuo.yan.2008@163.com

Received 20 February 2021; Revised 1 April 2021; Accepted 22 April 2021; Published 17 May 2021

Academic Editor: Zhipeng Cai

Nowadays, large volumes of multimodal data have been collected for analysis. An important type of data is trajectory data, which contains both time and space information. Trajectory analysis and clustering are essential to learn the pattern of moving objects. Computing trajectory similarity is a key aspect of trajectory analysis, but it is very time consuming. To address this issue, this paper presents an improved branch and bound strategy based on time slice segmentation, which reduces the time to obtain the similarity matrix by decreasing the number of distance calculations required to compute similarity. Then, the similarity matrix is transformed into a trajectory graph and a community detection algorithm is applied on it for clustering. Extensive experiments were done to compare the proposed algorithms with existing similarity measures and clustering algorithms. Results show that the proposed method can effectively mine the trajectory cluster information from the spatiotemporal trajectories.

1. Introduction

Nowadays, a huge amount of data is collected and it is important to develop tools to analyze data to extract useful knowledge. The collected data is often multimodal, that is of different types (e.g., audio [1], video [2], text [3], and image [4]), and can be analyzed jointly or separately [5, 6]. An emerging type of data that is playing a key role in multimodal data analysis is trajectory data [7]. It consists of spatial and temporal information about moving objects. Common trajectory data can be divided into four categories, namely, human trajectories, vehicle trajectories, animal trajectories, and natural phenomenon trajectories. Analyzing and discovering patterns in trajectory data have applications in several fields such as intelligent transportation, human mobility analysis, urban planning, meteorology, and travel recommendations and can reveal insights that are not discovered from other data types.

The process of trajectory data analysis mainly consists of obtaining and preprocessing trajectory data, trajectory data management, and a variety of mining tasks, including trajectory pattern mining, privacy protection, outlier detection [8, 9], and clustering trajectories on complex road networks [10, 11]. Many studies have been published, and trajectory data analysis is a very active research field. A generative adversarial network (GAN) was used to predict pedestrian movement by analyzing multimodal trajectory data [12]. However, most techniques for trajectory data analysis require measuring trajectory similarity, which necessitates a large amount of calculations on trajectory data and results that the time complexity of these similarity measurement methods is relatively high. Based on the idea of branch and bound, a novel similarity measurement method, called FSTM [13], was proposed that sets a distance threshold to prune certain mismatched points. Still, FTSM only considers space constraints.

More recently, there is an increasing interest on time series clustering using graphs [14, 15]. Traditional analysis methods only focus on the local relationship between data samples, while ignoring the global information. Advanced trajectory data mining techniques take network dynamics of trajectories into account, such as to mine trajectory group patterns and to assess the importance of a moving object in trajectory networks [16–18]. A complex network is suitable for revealing important relationships in trajectory data visually and can provide global information as time series data. In addition, there is no restriction on the shape of clusters.

Based on the above advantages and limitations, we propose an approach to spatiotemporal trajectory clustering based on community detection (*STTC-CD*). The algorithm implements an improved branch and bound strategy based on time slice segmentation. While richer trajectory information is taken into consideration, redundant trajectory points are pruned. Then, the trajectory data is converted into a graph representation based on the similarity matrix. Finally, a suitable community detection algorithm is applied to perform clustering on the graph. The main contribution of this paper is as follows:

(i) An improved similarity calculation method is designed, which matches pairs of trajectory points and applies a pruning strategy based on time slicing to reduce the time complexity

(ii) A method is proposed to convert trajectories into a suitable data format to apply many types of techniques for trajectory data mining. Based on this, a community detection algorithm is applied to cluster trajectories, which captures global relationships among trajectories from a graph-based perspective

(iii) Experiments have been conducted to evaluate the proposed algorithm on several datasets to verify the influence of multiple factors. It was found that the proposed algorithm is more efficient than the compared methods

The rest of this paper is organized as follows: Section 2 surveys relevant related work. Section 3 formally defines the trajectory clustering problem. Section 4 presents the designed *STTC-CD* algorithm. Then, Section 5 describes the experimental evaluation and Section 6 draws a conclusion.

2. Related Work

The key problem in trajectory clustering is how to measure trajectory similarity. This section first reviews techniques for trajectory similarity measurement and then surveys relevant work on community detection.

2.1. Trajectory Similarity Measure. Most trajectory data analysis tasks require computing trajectory similarity measurements, such as trajectory clustering [19], transforming data for privacy-preservation [20], movement pattern mining [21], and abnormal trajectory detection [22]. Traditional trajectory measurement techniques such as EDR (edit distance on real sequence), LCSS (longest common subsequence), and DTW (Dynamic Time Warping) compute the overall trajectory similarity by analyzing each trajectory as a whole rather than considering subtrajectories or random trajectory points. Among these techniques, DTW [23] aligns trajectories of different lengths by warping a trajectory sequence and can match a point at a certain time from a trajectory to a number of continuous points from another trajectory. Hence, it has no restriction on the length of the compared trajectories. LCSS [24] calculates the longest common subsequence of two trajectories as their similarity. EDR calculates the minimum number of changes required to transform a trajectory into another as the similarity between the two trajectories. Clue-Aware Trajectory Similarity (CATS) [25] is aimed at overcoming the influence of track bias in time and space. Multidimensional Similarity Measure (MSM) [26] and Multiple-Aspect Trajectory Similarity Measure (MUITAS) [27] provide similarity measures for multidimensional sequences, adding information such as weather, user activity, and user interest into trajectory comparison.

However, DTW is a distance-based method, which directly accumulates the distances between trajectory point pairs. A problem of DTW is that the sum of the distances can greatly increase when there are noise points, which makes it sensitive to noise points. Quite the reverse, the ε -threshold-based measures use an ε-threshold value to determine if two points match, which can be more robust to noise. LCSS, EDR, CATS, and MSM fall all in the ε-threshold-based strategy, and the computation of similarity score is based on the point matching of two trajectories. They have a $O(n^2)$ time complexity and cause a performance bottleneck for trajectory clustering algorithms. Furtado et al. proposed a branch and bound method (FTSM) to achieve fast similarity measuring by utilizing a transitive range pruning strategy to reduce the number of matching point pairs.

2.2. Community Detection in Networks. A community is a subset of network nodes. Connections between nodes within a subset are relatively close, while connections between nodes from different subsets are relatively sparse, which is exactly in line with the needs and principles of clustering. Recently, community detection algorithms have been increasingly utilized for trajectory clustering.

Depending on whether a node can belong to multiple communities or only one, community detection methods can be categorized as finding nonoverlapping or overlapping communities. In a nonoverlapping community, each network node can belong to one community. Algorithms that detect communities of this type are Fastgreedy [28], Louvain [29], Label Propagation [30], and Infomap [31]. Modularity is used to measure the quality of community division. The Fastgreedy algorithm applies a bottom-up process. Initially, each node is regarded as a community. Then, at each iteration, the two communities providing the largest increase in modularity are merged until the entire network is merged into a single community. The final community structure is a division that maximizes the modularity. The Louvain algorithm improves upon the Fastgreedy algorithm by assigning each node to neighboring nodes for maximum modularity. When the ownership of a node no longer changes, the algorithm collapses each community into a node to form a new community for the next iteration. The basic idea of the Label Propagation algorithm (LPA) is to predict labels of unlabeled network nodes from labeled nodes. Each node label is propagated to neighboring nodes according to their similarity. At each step of node propagation, the node updates itself according to the label of the neighboring node until the label no longer changes. Similar to K-means, the results of LPA are affected by the initial label selection. The Infomap algorithm introduces a coding-based technique based on random walks. A good group division can lead to shorter coding length.

A trajectory clustering algorithm based on an improved Label Propagation algorithm was proposed where road network is modeled as a dual graph to capture and characterize the similarity between nodes [10]. Liu and Guo proposed a semantic trajectory clustering algorithm based on community detection [32], where different community detection algorithms were discussed.

3. Problem Statement

The following definitions are provided to facilitate the formulation of the problem under study:

Definition 1 (trajectory). A trajectory is a sequence of points in chronological order, denoted as $\mathrm{TR}_i = \{p_i^1, p_i^2, \cdots, p_i^j, \cdots, p_i^{n_i}\}$, where each point $p_i^j = (i, x, y, t)$ represents the spatial location (x, y) of an entity at given time t of trajectory TR_i, and n_i is the number of points in TR_i.

Definition 2 (silhouette coefficient SI). The silhouette coefficient is a metric to evaluate the quality of a clustering, which considers two aspects that are cohesion and resolution. The $s(i)$ of each trajectory point p_i is calculated as follows:

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}, \quad (1)$$

where $a(i)$ denotes the average distance from p_i to all trajectory points in the cluster to which p_i belongs, and $b(i)$ is the average distance between p_i and trajectory points in other clusters. Given a trajectory dataset $\mathrm{TS} = \{\mathrm{TR}_1, \mathrm{TR}_2, \cdots, \mathrm{TR}_N\}$, the silhouette coefficient of TS is the average of the silhouette coefficients of all trajectories, denoted as

$$\mathrm{SI} = \frac{1}{N}\sum_{j=1}^{N}\frac{1}{n_j}\sum_{i=1}^{n_j} s(i), \quad (2)$$

where N is the number of trajectories, n_j is the number of trajectory points in TR_j, and $(1/n_j)\sum_{i=1}^{n_j}\boxtimes s(i)$ is the silhouette coefficient of trajectory TR_j.

The value of SI is between -1 and 1 such that a higher SI value indicates a better clustering result in general. According to the above definition, the road trajectory clustering optimization problem is defined as follows:

Definition 3 (trajectory clustering optimization problem). Given a set of trajectories $\mathrm{TS} = \{\mathrm{TR}_1, \mathrm{TR}_2, \cdots, \mathrm{TR}_N\}$ in Euclidean space for the time period $[0, T]$, the goal is to divide TS into groups $\{C_1, C_2, \cdots, C_{N_c}\}$ to maximize SI.

4. The Proposed STTC-CD Algorithm

This paper proposes an approach to spatiotemporal trajectory clustering based on community detection, named *STTC-CD*, which is applied in three steps: (1) trajectory partition, (2) graph generation, and (3) trajectory clustering, as illustrated in Figure 1.

Stage 1. Trajectory Partition. Given a collection of space-time trajectories $\{\mathrm{TR}_1, \mathrm{TR}_2, \cdots, \mathrm{TR}_N\}$, *STTC-CD* divides them into time slices and then utilizes transitive range pruning to calculate the number of pairs of matching points between trajectories in each time period to generate a matching matrix.

Stage 2. Graph Generation. STTC-CD aggregates the matching matrix of each time period to generate a global matching matrix. Then, the algorithm transforms the matching matrix into a similarity matrix according to similarity rules, and a trajectory-connected graph is generated.

Stage 3. Trajectory Clustering. Based on the trajectory graph obtained in the second stage, we utilize a community detection algorithm for clustering to capture global relationships between trajectories from the perspective of the network.

4.1. Trajectory Partition. An algorithm is proposed that takes the time characteristics of trajectories into account and utilizes a branch and bound strategy for fast trajectory similarity measurement. The algorithm is called *STTC-CD*. It not only improves the accuracy of similarity measurement but also only compares each trajectory segment with others from the same time slice instead of all trajectories, thereby improving computational efficiency through further pruning.

Given a trajectory dataset $\mathrm{TS} = \{\mathrm{TR}_1, \mathrm{TR}_2, \cdots, \mathrm{TR}_N\}$ and a partition threshold κ, TS is divided into κ subdatasets $\{\mathrm{TS}_1, \mathrm{TS}_2, \cdots, \mathrm{TS}_\kappa\}$ according to the time slice and then allocated to the corresponding subdataset of the time slice. Let $t_{\min}$ and $t_{\max}$ be the minimum and maximum timestamp in the dataset, respectively. The length of each time slice is defined as follows:

$$\Delta t = \frac{t_{\max} - t_{\min}}{\kappa}. \quad (3)$$

Each trajectory $\mathrm{TR}_i = \{p_i^1, p_i^2, \cdots, p_i^m, \cdots, p_i^{n_i}\} \in \mathrm{TS}$ is divided into subdatasets according to the time slice (as shown in Figure 2). The index of the subdataset to which a point p_i^m is assigned is $\lceil (t_i^m - t_{\min})/\Delta t \rceil$.

4.2. Graph Generation. The graph is generated based on the similarity matrix. The calculation of similarity in each time slice is done based on the following definitions:

Definition 4 (point matching (PM)). Let there be two points p_i and p_j, a matching threshold ε, and a distance function $\mathrm{dist}(p_i, p_j)$. If $\mathrm{dist}(p_i, p_j) \le \varepsilon$, then p_i and p_j match each other; otherwise, they do not match. The formula is defined as follows:

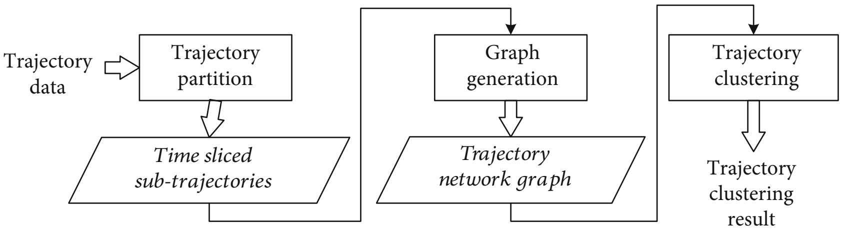


Figure 1: Algorithm flowchart.

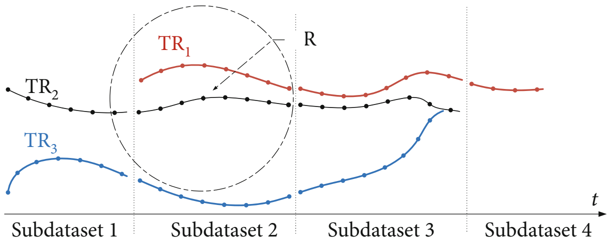


Figure 2: The schematic diagram of trajectory partition.

$$\mathrm{PM} = \begin{cases} 1, & \mathrm{dist}\left(p_i, p_j\right) \le \varepsilon, \\ 0, & \mathrm{dist}\left(p_i, p_j\right) > \varepsilon. \end{cases} \tag{4}$$

Definition 5 (trajectory segment matching (TM)). Given two trajectory segment $\mathrm{sTR}_i = \{p_1, p_2, \cdots, p_m\}$ and $\mathrm{sTR}_j = \{q_1, q_2, \cdots, q_n\}$, trajectory segment matching is defined as follows:

$$\mathrm{TM}\left(\mathrm{sTR}_i, \mathrm{sTR}_j\right) = \sum_{i=1}^{m}\sum_{j=1}^{n} \mathrm{PM}\left(p_i, q_j\right) + \sum_{j=1}^{n}\sum_{i=1}^{m} \mathrm{PM}\left(q_j, p_i\right), \tag{5}$$

where m and n are the numbers of the points of sTR_i and sTR_j.

Considering that trajectory elements are points in Euclidean space, the following definitions adopt the Euclidean distance as distance function to perform point matching. Hence, the matching threshold can be seen as the radius ε of a matching circle.

Definition 6 (pivot point). For a trajectory TR_i, the pivot point of TR_i is the point at half of the trajectory as follows:

$$p_i^k = p_i^{\lfloor (1+n_i)/2 \rfloor}, \tag{6}$$

where n_i is the number of trajectory points of TR_i.

Definition 7 (pruning radius (PR)). Given a pivot point $p_i^k \in \mathrm{TR}_i$ and a matching threshold ε, the pruning radius is a circle around p_i^k that covers all the points that are at maximum distance ε of any point in TR_i, that is,

$$\mathrm{PR} = \varepsilon + \max\left(\sum_{m=1}^{k-1} \mathrm{dist}\left(p_i^m, p_i^{m+1}\right), \sum_{n=k}^{n_i-1} \mathrm{dist}\left(p_i^n, p_i^{n+1}\right)\right). \tag{7}$$

Lemma 8 (transitive range pruning). *Let* $TR_i = \{p_i^1, p_i^2, \cdots, p_i^m, \cdots, p_i^{n_i}\}$ *and* $TR_j = \{p_j^1, p_j^2, \cdots, p_j^n, \cdots, p_j^{n_j}\}$ *be two trajectories,* ε *be a matching threshold,* $dist(p_i^m, p_j^n)$ *be the metric computing the distance between two points, and PR be a pruning radius around a pivot point* $p_i^k \in TR_i$*. Then, for any point* $p_i^m \in TR_i$ *and* $p_j^n \in TR_j$,

$$dist\left(p_j^n, p_i^m\right) \le \varepsilon \Rightarrow dist\left(p_j^n, p_i^k\right) \le PR. \tag{8}$$

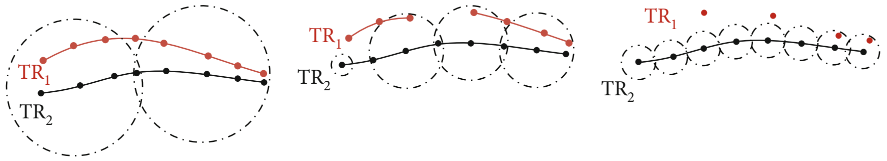
This lemma [13] means that for any point in TR_j, if its distance to a certain point of TR_i is less than ε, then its distance to the pivot point of TR_i must be less than PR. Therefore, if the distance from a point to the pivot point of TR_i is greater than PR, the distance from it to all points of TR_i is greater than ε, and the pruning operation can be performed accordingly.

Based on the subdatasets generated in Stage 1, the number of matching points in each subdataset is calculated. Given two subtrajectories sTR_i and sTR_j, the calculation of point matching consists of three steps, as shown in Figure 3:

(a) Pruning step: the pivot point of sTR_i is denoted as p_i^k. For any point $p_j^m \in \mathrm{sTR}_j$, the distance is calculated from p_i^k to p_j^m and is compared with the threshold PR. If $\mathrm{dist}(p_i^k, p_j^m)\mathrm{PR}$, p_j^m is added to the matching queue

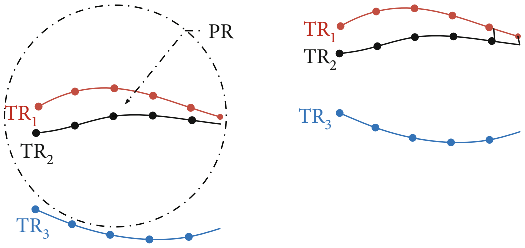
(b) Splitting step: sTR_i is separated from the pivot point p_i^k to form two subtrajectories. The center points of subtrajectories are taken as new pivot points, and the points in the matching queue form the new sT R$_j$. The pruning step is repeated until the matching queue is empty or the trajectory segment can no longer be divided

(c) Matching step: the points of sTR_j in the matching queue are matched with sTR_i to get the number of matching points



(a) (b) (c)

Figure 3: The schematic diagram of point matching.



(a) Subdataset 2 (b) Matching result of subdataset 2

Figure 4: Matching step for a subdataset.

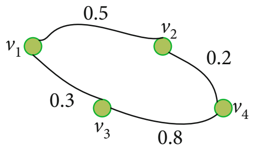


Figure 5: Trajectory graph.

Table 1: Datasets.

DS#	Dataset	Trajectory count	Avg. trajectory points	Time span
DS1	Trucks	1,100	85	39 days
DS2	T-drive	10,357	1448	7 days
DS3	UCI	163	111	493 days

For instance, Figure 4 shows one of the subdatasets after partition. Figure 4(a) is a subdataset consisting of three sub-trajectories, and Figure 4(b) shows the matching result of it, where the number of matching points between sTR$_2$ and other trajectories in subdataset 2 is calculated as 2 and 0.

The matching matrix is aggregated of each time slice. According to the matching point matrix, the similarity matrix can be obtained. The similarity is defined as follows:

Definition 9 (trajectory similarity measure). For two trajectories $\mathrm{TR}_1 = \{\mathrm{sTR}_1^{(1)}, \mathrm{sTR}_1^{(2)}, \cdots, \mathrm{sTR}_1^{(m)}\}$ and $\mathrm{TR}_2 = \{\mathrm{sTR}_2^{(1)}, \mathrm{sTR}_2^{(2)}, \cdots, \mathrm{sTR}_2^{(n)}\}$, the similarity of them is calculated as

$$\mathrm{Sim}\left(\mathrm{TR}_i, \mathrm{TR}_j\right) = \mathrm{Sim}\left(\mathrm{TR}_j, \mathrm{TR}_i\right) = \frac{\sum_{p=1}^{m}\sum_{q=1}^{n}\mathrm{TM}\left(\mathrm{sTR}_i^{(p)}, \mathrm{sTR}_j^{(q)}\right) + \sum_{q=1}^{n}\sum_{p=1}^{m}\mathrm{TM}\left(\mathrm{sTR}_j^{(q)}, \mathrm{sTR}_i^{(p)}\right)}{m+n}, \quad (9)$$

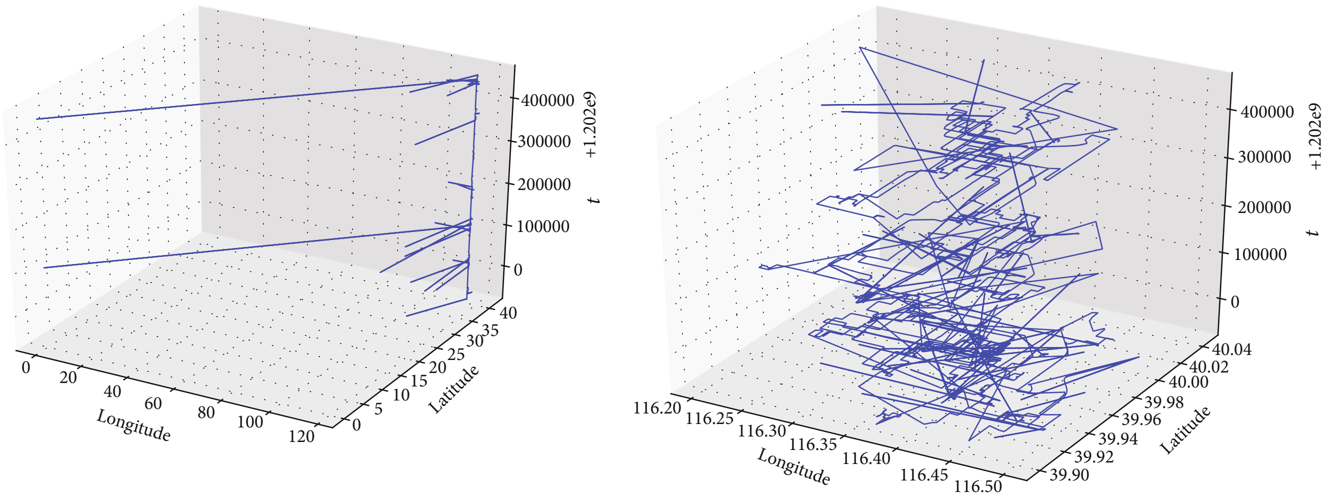
where m and n are the number of sub-trajectories in TR$_i$ and TR$_j$, respectively. The similarity measure satisfies the property of nonnegativity, which means $\mathrm{Sim}(\mathrm{TR}_i, \mathrm{TR}_j) \geq 0$ in all cases, and a large score indicates a high similarity.

Then, the matching matrix is transformed by Equation (9) to obtain the similarity matrix S, where $\mathrm{Sim}(\mathrm{TR}_i, \mathrm{TR}_j)$ represents the similarity between TR$_i$ and TR$_j$. A trajectory graph $G = (V, E)$ is constructed by exploiting the similarity matrix S. Firstly, N vertices are constructed for a dataset with N trajectories and each trajectory corresponds to a vertex. For each v_i corresponding to the trajectory TR$_i$ and v_j corresponding to the trajectory TR$_j$, edge is added between them if $\mathrm{Sim}(\mathrm{TR}_i, \mathrm{TR}_j) > 0$. The weight of each edge is equal to the similarity between the two vertices. For instance, given a matrix $[[0,0.5,0.3,0], [0.5,0, 0,0.2], [0.3,0, 0,0.8], [0,0.2,0.8,0]]$, the trajectory graph is as shown in Figure 5.

4.3. *Trajectory Clustering.*

A community is composed of a group of closely connected nodes that are sparsely connected with nodes outside the community. Community detection is to discover these closely connected community structures in a complex network, which coincides with the objective of clustering. Therefore, the Infomap algorithm [31] is employed for clustering, which combines community detection with information encoding.

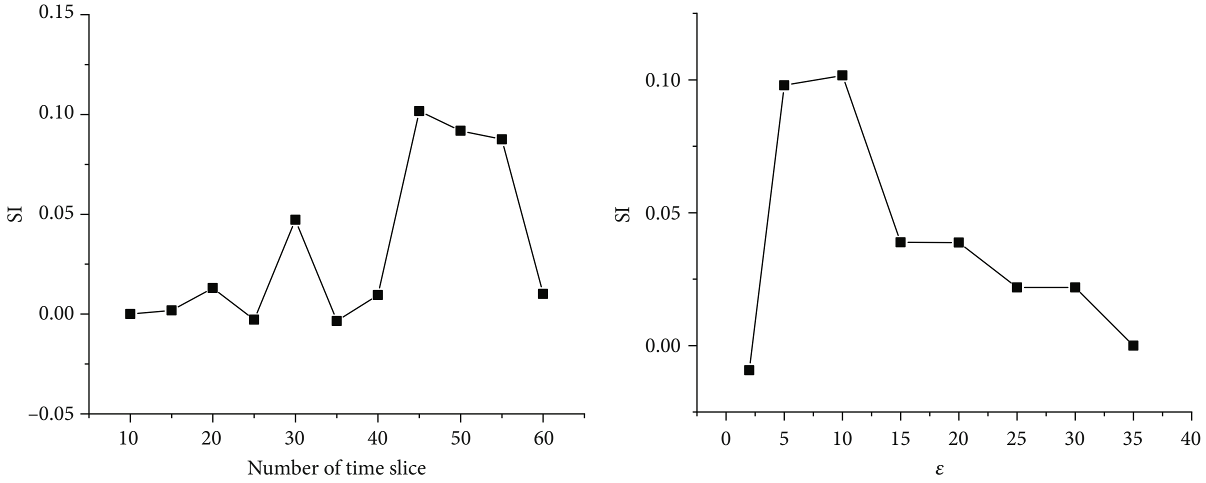
The basic idea of the Infomap algorithm is to find the shortest codes to describe the path generated by a random walk on the network. This is done using a two-level coding of all network nodes to find the module partition with the shortest encoding length by minimizing entropy to find the optimal clustering. The two-level code assigns unique module names, and nodes in different modules are allowed to use repeated codewords. The module code is inserted before the nodes in the same module, and the termination mark is inserted at the end. The average code



(a) Original trajectory 6275

(b) Processed trajectory 6275

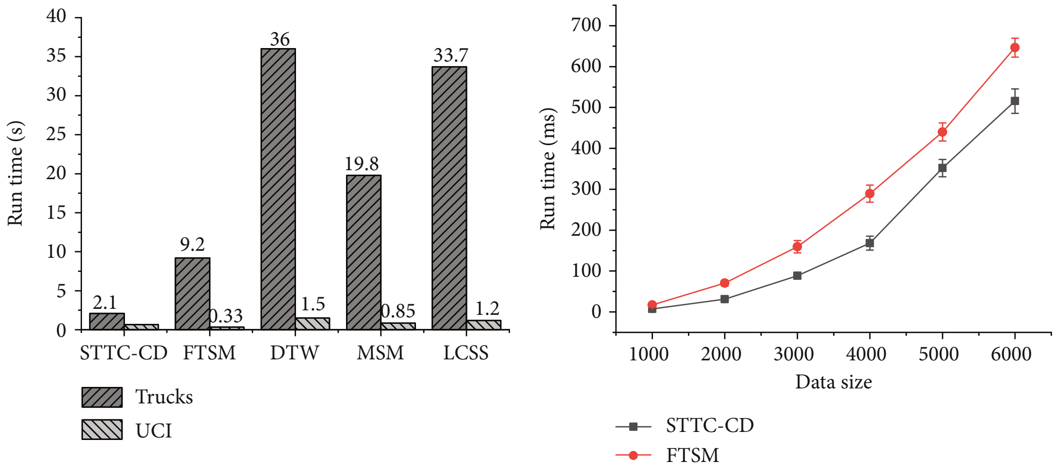
Figure 6: Comparison graph of trajectory processing.



(a) Different time slice (ε = 10)

(b) Different ε (time slice = 45)

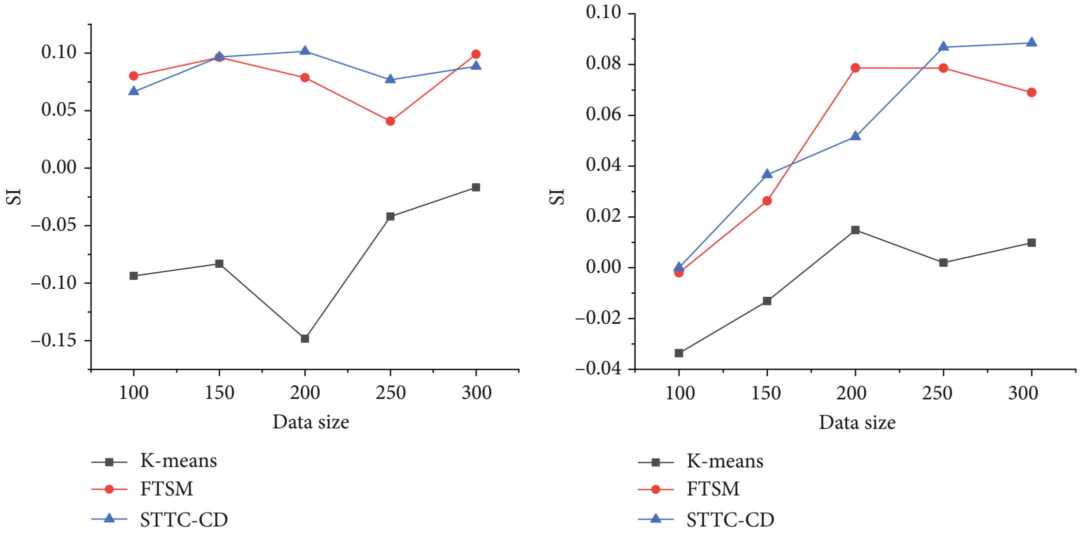
Figure 7: Clustering quality on trucks dataset using SI with different parameters.



(a) Runtime on DS1 and DS3

(b) Runtime on DS2

Figure 8: Runtime.



(a) SI index on DS1

(b) SI index on DS2

Figure 9: SI.

length is calculated as follows:

$$L(M) = q \Subset H(Q) + \sum_i p^i_O H(P^i), \tag{10}$$

where $q \Subset$ represents the probability of switching from one module to another per step of the random walk, $H(Q)$ is the entropy of movements between modules, p^i_O denotes the proportion of all nodes in group i in the encoding, and $H(P^i)$ denotes the average code length required by all nodes in group i. The Infomap algorithm performs three steps:

Step 1. Initialization. Each graph node is treated as an independent group.

Step 2. Each node is traversed in a random order, and each point is assigned to the adjacent module that gives the largest decrease in Equation (10).

Step 3. Step 2 is repeated in a different random order until Equation (10) does not decrease.

5. Performance Evaluation

The performance of the proposed SSTC-CD algorithm was evaluated in terms of silhouette coefficient and runtime. All algorithms were implemented in Java 14, and all experiments were conducted on a Windows PC workstation equipped with an Intel(R) Core(TM) i5-10400 CPU@2.90 GHz and 16 GB of memory.

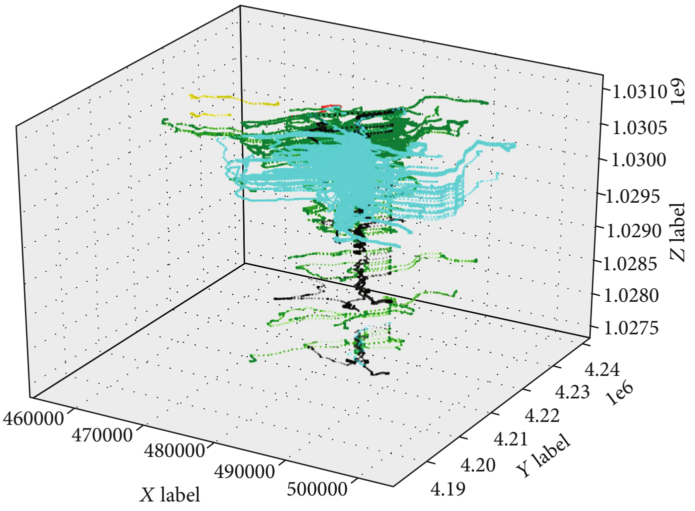
5.1. Datasets. The algorithm was evaluated on several widely used public datasets, described in Table 1. The trucks dataset (DS1) is a real-word dataset composed of 1,100 trajectories generated by 50 different trucks transporting concrete in Greece. T-drive dataset [33] (DS2), provided by Microsoft Research Asia, is a collection of trajectories generated by 10,357 taxis located in Beijing within a week. The UCI dataset (DS3) was collected by the GoTrack Android app in 2016. It has a high sampling rate for a single trajectory, but the interval between trajectories is long.

DS2 was collected in Beijing, which is located in longitude 115.7°E to 117.4°E and latitude 39.4°N to 41.6°N. Therefore, out-of-range points were deleted as abnormal points. The average trajectory length in DS2 is about 1,500 points. Yet, the longest trajectory has 150,000 points, and there are many repeated points and stay points, which we have removed from the dataset. Figure 6 presents the longest trajectory in DS2 with id 6275. Figure 6(a) is the original trajectory, and Figure 6(b) is the processed trajectory.

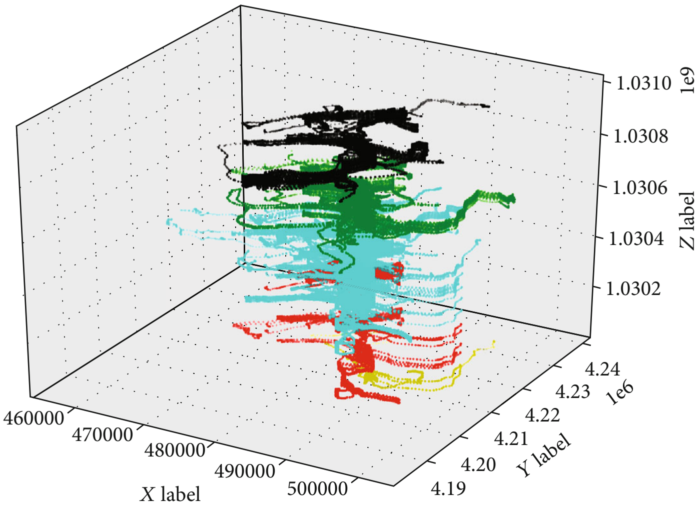
5.2. Evaluation. In our experiments, we run STTC-CD with different ε-threshold and different number of time slices to identify the optimal parameters. Figure 7 shows the influence of different parameters on the proposed algorithm. As shown in Figure 7(a), the SI index shows a trend of rising first and then falling as the number of time slices increases, and it reaches the maximum value when the number of time slices is 45. As shown in Figure 7(b), the value of ε was set from 2 to 35 and the SI index reaches its maximum value when ε is 10.

The performance of the proposed *STTC-CD* algorithm was compared with several similarity measurement algorithms, namely, FTSM [13], DTW [23], MSM [26], and LCSS [24], on DS1 and DS3. The parameter ε was set to 10, and the number of time slices was set to 45. Results are presented in Figure 8(a).

It can be observed that the running time of *STTC-CD* and FTSM on both datasets is shorter than that of other algorithms. For large datasets, the runtime gap is greater. The reason is that the other three algorithms are implemented using dynamic programming, which have quadratic time complexity. As the data size increases, the time required by these



(a) FTSM clustering results on DS1



(b) *STTC-CD* clustering results on DS1

Figure 10: Clustering result on DS1.

algorithms rises sharply. Since FTSM and *STTC-CD* pruned the sequence to be matched on the trajectory, the complexity is close to linear in the best case. When the data size is small, *STTC-CD* prunes more pair-wise trajectory points than FTSM by splitting in time slices. However, the operations of splitting and matching time slices take more time, which results in spending more time than FTSM.

To further evaluate FTSM and *STTC-CD*, DS2 was split into six subdatasets of different sizes and the two algorithms were applied. It can be seen in Figure 8(b) that when the dataset is small, the runtimes of the two algorithms are almost the same. As dataset size increases, the gap becomes more obvious. This result is also consistent with the results for the other two datasets.

The performance of algorithms was further compared in terms of the SI index. The time dimension of the dataset is considered in the algorithm; therefore, the three-dimensional Euclidean distance combined with the time dimension is utilized as the distance measure of SI. Compared with DTW, MSM, and LCSS implemented by dynamic programming, FTSM only pruned away some unnecessary comparisons, which improved the running speed of the algorithm without affecting the accuracy of the algorithm. Based on FTSM, the proposed algorithm further reduces the number of point matching in similarity calculation, but it also affects the accuracy of the algorithm. Therefore, the SI index was used to compare the accuracy of FTSM and STTC-CD, and K-means was used as the benchmark algorithm. As illustrated in Figure 9, the proposed algorithm was compared with FTSM and K-means with different numbers of trajectories on DS1 and DS2. It can be observed that the SI of FTSM and *STTC-CD* are greater than the SI of K-means on both datasets, and most of the time, *STTC-CD* results are better than FTSM, which indicates that the proposed *STTC-CD* takes better account of time correlation.

The clustering results of FTSM and *STTC-CD* on DS1 are displayed using lines of different colors, while trajectories from the same cluster are represented using the same color. Figure 10(a) shows the clustering result of FTSM, and Figure 10(b) shows the clustering result of *STTC-CD*. It is found that FTSM does not discriminate in the time dimension. In contrast, the proposed algorithm has better results in the division of time levels.

6. Conclusion

This article presented an approach to spatiotemporal trajectory clustering based on community detection (*STTC-CD*), which is based on time slicing to reduce the time for similarity calculation. *STTC-CD* relies on a new trajectory representation, which enables various algorithms such as for community detection to be applied for trajectory clustering. Experimental results have shown that the proposed algorithm can effectively reduce runtimes on large datasets and that clustering results are more meaningful in the time dimension.

The approach proposed in this paper is designed to analyze and cluster trajectory data. An interesting research possibility for future work is to see this work as a building block to build a system for analyzing multimodal data consisting not only of trajectory but also text, video, and audio data. In particular, a hybrid system could be developed combining the proposed approach with a neural network or other machine learning models.

Data Availability

The T-drive dataset used to support the findings of this study has been deposited in the Microsoft Research Asia (doi:10.1145/2020408.2020462). The trucks dataset used to support the findings of this study is included within the article "Clustering Trajectories of Moving Objects in an Uncertain World" (doi:10.1109/ICDM.2009.57). The UCI dataset used to support the findings of this study has been feed by Android app called GoTrack. It is available at Google Play Store.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research is sponsored by the Key Research and Development Program under Grant No. 2020YFS0169, Science and Technology Department of Sichuan Province and the Science and Technology Planning Project of Sichuan Province under Grant No. 2020YFG0054, and the Joint Funds of the Ministry of Education of China.

References

[1] A. Crossan and S. Brewster, "Multimodal trajectory playback for teaching shape information and trajectories to visually impaired computer users," *ACM Transactions on Accessible Computing*, vol. 1, no. 2, pp. 1–34, 2008.

[2] C. Hori, T. Hori, T. Lee et al., "Attention-based multimodal fusion for video description," in *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 4203–4212, Venice, Italy, 2017.

[3] B. Qiang, R. Chen, Y. Xie, M. Zhou, R. Pan, and T. Zhao, "Hybrid deep neural network-based cross-modal image and text retrieval method for large-scale data," *Journal of Circuits, Systems and Computers*, vol. 30, 2020.

[4] A. R. Groves, C. F. Beckmann, S. M. Smith, and M. W. Woolrich, "Linked independent component analysis for multimodal data fusion," *NeuroImage*, vol. 54, no. 3, pp. 2198–2217, 2011.

[5] D. Lahat, T. Adal, and C. Jutten, "Multimodal data fusion: an overview of methods, challenges and prospects," *Proceedings of the IEEE*, vol. 103, no. 9, pp. 1449–1477, 2015.

[6] J. Gao, P. Li, Z. Chen, and J. Zhang, "A survey on deep learning for multimodal data fusion," *Neural Computation*, vol. 32, no. 1, pp. 1–36, 2020.

[7] Z. Zhang, J. Rhim, M. TaherAhmadi, K. Yang, A. Lim, and M. Chen, "Sfu-store-nav: a multimodal dataset for indoor human navigation," *Data in Brief*, vol. 33, p. 106539, 2020.

[8] P. Peltola, J. Xiao, T. Moore, A. Jiménez, and F. Seco, "Gnss trajectory anomaly detection using similarity comparison methods for pedestrian navigation," *Sensors*, vol. 18, no. 9, article 3165, 2018.

[9] Z. Fu, W. Hu, and T. Tan, "Similarity based vehicle trajectory clustering and anomaly detection," in *IEEE International Conference on Image Processing*, vol. 2, pp. 602–605, Genova, Italy, 2005.

[10] X. Niu, T. Chen, C. Q. Wu, J. Niu, and Y. Li, "Label-based trajectory clustering in complex road networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 10, pp. 4098–4110, 2020.

[11] J. Zhu, X. Niu, and C. Q. Wu, "On a clustering-based approach for traffic sub-area division," in *Advances and Trends in Artificial Intelligence*, pp. 516–529, Springer, 2019.

[12] J. Amirian, J. B. Hayet, and J. Pettre, "Social ways: learning multi-modal distributions of pedestrian trajectories with gans," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 2964–2972, Long Beach, USA, 2019.

[13] A. S. Furtado, L. L. Pilla, and V. Bogorny, "A branch and bound strategy for fast trajectory similarity measuring," *Data & Knowledge Engineering*, vol. 115, pp. 16–31, 2018.

[14] M. A. Ghamdi and Y. Gotoh, "Graph-based topic models for trajectory clustering in crowd videos," *Machine Vision and Applications*, vol. 31, no. 5, 2020.

[15] B. Sabarish, R. Karthi, and T. G. Kumar, "Graph similarity-based hierarchical clustering of trajectory data," *Procedia Computer Science*, vol. 171, pp. 32–41, 2020.

[16] T. Pechlivanoglou and M. Papagelis, "Fast and accurate mining of node importance in trajectory networks," in *2018 IEEE International Conference on Big Data (Big Data)*, pp. 781–790, Seattle, WA, USA, 2018.

[17] L. Zhao and G. Shi, "A trajectory clustering method based on Douglas-Peucker compression and density for marine traffic pattern recognition," *Ocean Engineering*, vol. 172, pp. 456–467, 2019.

[18] V. Mirge, K. Verma, and S. Gupta, "Dense traffic flow patterns mining in bi-directional road networks using density based trajectory clustering," *Advances in Data Analysis and Classification*, vol. 11, no. 3, pp. 547–561, 2017.

[19] G. Yuan, P. Sun, J. Zhao, D. Li, and C. Wang, "A review of moving object trajectory clustering algorithms," *Artificial Intelligence Review*, vol. 47, no. 1, pp. 123–144, 2017.

[20] O. Abul, F. Bonchi, and M. Nanni, "Never walk alone: uncertainty for anonymity in moving objects databases," in *IEEE International Conference on Data Engineering*, pp. 376–385, Cancun, Mexico, 2008.

[21] K. Zheng, Y. Zheng, N. J. Yuan, S. Shang, and X. Zhou, "Online discovery of gathering patterns over trajectories," *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 8, pp. 1974–1988, 2014.

[22] X. Zhao, Y. Rao, J. Cai, and W. Ma, "Abnormal trajectory detection based on a sparse subgraph," *IEEE Access*, vol. 8, pp. 29987–30000, 2020.

[23] D. J. Berndt and J. Clifford, "Using dynamic time warping to find patterns in time series," in *Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining*, pp. 359–370, Seattle, Washington, 1994.

[24] M. Vlachos, D. Gunopoulos, and G. Kollios, "Discovering similar multidimensional trajectories," in *Proceedings of the 18th International Conference on Data Engineering*, p. 673, San Jose, CA, USA, 2002.

[25] C. C. Hung, W. C. Peng, and W. C. Lee, "Clustering and aggregating clues of trajectories for mining trajectory patterns and routes," *The VLDB Journal*, vol. 24, no. 2, pp. 169–192, 2015.

[26] A. S. Furtado, D. Kopanaki, L. O. Alvares, and V. Bogorny, "Multidimensional similarity measuring for semantic trajectories," *Transactions in GIS*, vol. 20, no. 2, pp. 280–298, 2016.

[27] L. M. Petry, C. A. Ferrero, L. O. Alvares, C. Renso, and V. Bogorny, "Towards semantic-aware multiple-aspect trajectory similarity measuring," *Transactions in GIS*, vol. 23, no. 5, pp. 960–975, 2019.

[28] M. E. Newman, "Fast algorithm for detecting community structure in networks," *Physical Review. E, Statistical, Nonlinear, and Soft Matter Physics*, vol. 69, no. 6, article 066133, 2004.

[29] V. D. Blondel, J. L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2008, article P10008, no. 10, 2008.

[30] U. N. Raghavan, R. Albert, and S. Kumara, "Near linear time algorithm to detect community structures in large-scale networks," *Physical Review E*, vol. 76, no. 3, article 036106, 2007.

[31] M. Rosvall and C. Bergstrom, "Maps of random walks on complex networks reveal community structure," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 105, no. 4, pp. 1118–1123, 2008.

[32] C. Liu and C. Guo, "Stccd: semantic trajectory clustering based on community detection in networks," *Expert Systems with Applications*, vol. 162, p. 113689, 2020.

[33] Y. Zheng, "T-drive trajectory data sample," 2011, t-Drive sample dataset, https://www.microsoft.com/en-us/research/publication/t-drive-trajectory-data-sample/.

Hindawi
Wireless Communications and Mobile Computing
Volume 2021, Article ID 5514743, 10 pages
https://doi.org/10.1155/2021/5514743

Research Article

Prediction of HFMD Cases by Leveraging Time Series Decomposition and Local Fusion

Ziyang Wang,[1] Zhijin Wang ,[1] Yingxian Lin ,[1] Jinming Liu,[1] Yonggang Fu,[1] Peisong Zhang,[2] and Bing Cai[1]

[1] *Computer Engineering College, Jimei University, Xiamen 361021, China*
[2] *School of Science, Jimei University, Xiamen 361021, China*

Correspondence should be addressed to Zhijin Wang; zhijin@jmu.edu.cn and Yingxian Lin; yxlin@jmu.edu.cn

Received 24 February 2021; Revised 30 March 2021; Accepted 21 April 2021; Published 13 May 2021

Academic Editor: Philippe Fournier-Viger

Hand, foot, and mouth disease (HFMD) is an infection that is common in children under 5 years old. This disease is not a serious disease commonly, but it is one of the most widespread infectious diseases which can still be fatal. HFMD still poses a threat to the lives and health of children and adolescents. An effective prediction model would be very helpful to HFMD control and prevention. Several methods have been proposed to predict HFMD outpatient cases. These methods tend to utilize the connection between cases and exogenous data, but exogenous data is not always available. In this paper, a novel method combined time series composition and local fusion has been proposed. The Empirical Mode Decomposition (EMD) method is used to decompose HFMD outpatient time series. Linear local predictors are applied to processing input data. The predicted value is generated via fusing the output of local predictors. The evaluation of the proposed model is carried on a real dataset comparing with the state-of-the-art methods. The results show that our model is more accurately compared with other baseline models. Thus, the model we proposed can be an effective method in the HFMD outpatient prediction mission.

1. Introduction

Hand, foot, and mouth disease (HFMD) is a common infection caused by a group of viruses. It is likely to occur to children under 5 years old. HFMD causes a serious threat to children's health. Especially in developing Asian countries, this disease is more likely to cause big damage. China is a country with a large population and vast territory, and the development of different regions is uneven. Under this situation, it is difficult to control infectious diseases spread in China. HFMD has been a nationally notifiable disease since 2008. The new cases should be reported in 24 hours. However, the situation is still worsening. According to the data from the Chinese Centre for Disease Control and Prevention (CCDC) [1], nearly 2 million cases were reported in China in 2019, with an incident rate of over 137/100,000. Although most HFMD patients are self-limiting, HFMD can still be fatal. Thus, the prevention and control of HFMD are very important. And if health authorities had anticipated the situation before the outbreak, a lot of unnecessary damage could have been avoided.

Many methods have been proposed to predict HFMD cases. ARIMA is one of the most general time series models, which is already used in HFMD prediction work [2]. ARIMAX is the ARIMA with external parameters added, and study showed that the ARIMAX has better performance than ARIMA [3]. With the increase of computer computing power, multiple learning models are utilized in HFMD prediction, such as LSTM [4], RNN, and CNN-RNN [5]. These methods often attempt to learn the law of the disease spread trend based on a global predictor.

However, on the one hand, the HFMD outpatient data is nonlinear and nonstationary. On the other hand, the spread of HFMD is affected by complex and diverse external factors, such as climate, living habits, and living conditions. These two characteristics make it difficult to improve performance based on a global predictor. The relationship between target data and external factors provides a new idea to researchers, and many studies focus on prediction using external factors to enhance the model performance have been down. The data about external factors is named exogenous data to distinguish it from target data. In this paper, we use new ideas to improve the accuracy of prediction: time series decomposition and local fusion.

Essentially, decomposition is the process of dividing a complex problem into subproblems that can be easily solved. In our experiments, a classical method named Empirical Mode Decomposition (EMD) is used to decompose the HFMD outpatient data. This method decomposes a time series into several subseries named Intrinsic Mode Function (IMF) and a residual. Each IMF contains a local feature. In addition, in our study, the residual is also treated as an IMF. Each IMF is treated equally by local predictors in the experiment.

In this paper, we propose a Concurrent Autoregression with Decomposition (CARD) model for HFMD prediction. We try to improve the accuracy of prediction as much as possible without exogenous data. CARD generates predicted value by fusing the output of the local predictors. The method utilizes two linear autoregression predictors to process the past outpatient data and the IMFs, respectively. Then, a fusion component fuses the outputs of two linear predictors. Finally, a global predictor is introduced to generate the predicted result. In a word, we propose an effective time series decomposition and local fusion method, which can catch a higher accuracy than several general methods that only use history outpatient data.

The main contributions of this paper can be summarized as follows:

(1) We propose a novel prediction model, which applied time series decomposition and local fusion to the prediction of outpatient cases of HFMD

(2) A classical decomposition method named EMD is introduced to decompose the HFMD outpatient time series. Compare with several other decomposition methods, EMD is simpler and more efficient in this study

(3) The proposed method applies a linear weighted module to fuse the output of two local predictors. Each local predictor predicts an output result independently. Then, the fusion module trains to generate the final predicted value of the output of local predictors

The rest of the paper is organized as follows. Section 2 introduces related work. The CARD model we proposed is explained in detail in Section 3. Section 4 illustrates the experiment design. Section 5 analyzes the experimental results. Finally, the whole research is summarized in Section 6.

2. Related Work

This section introduces several most commonly used decomposition methods and fusion methods related to our research.

2.1. Decomposition Methods. A time series can be decomposed into several subseries via decomposition methods. For time series decomposition, the following methods are widely used: wavelet transform [6], Robust Seasonal-Trend Decomposition (RobustSTL) [7, 8], EMD [9], and Ensemble Empirical Mode Decomposition (EEMD) [10].

Wavelet transform [6] inherits and develops the idea of localization short-time Fourier transform. Wavelet transform is a local transform not only the frequency but also time can be obtained. The method replaces the basis of Fourier transform. For a signal that has been processed by wavelet transform, both frequency part and specific position in time can be obtained. Compared with Fourier transform, it has good time-frequency localization characteristics and can extract information on signals more effectively.

RobustSTL [7, 8] is a robust method for decomposing complex time series into trend, seasonality, and remainder components. This method allows for multiple seasonal, cyclic components, and multiple linear regressors with constant, flexible, seasonal, and cyclic influence.

EMD [9] is a Fourier transform-based signal decomposition method, which can process any nonlinear and nonstationary signal adaptively. Compared with most of the decomposition methods, EMD is easy to use, since EMD decomposes data based on the local feature of the data, so this method is adaptive and does not require setting up extra parameters in advance.

EEMD [10] is a variant of EMD. For EMD, the extremum points of the signal will affect IMFs, and mode mixing will occur if the distribution of the extremum is uneven. EEMD is proposed to solve the mode mixing problem of EMD. This method using the advantage of uniform distribution of white noise spectrum, the white noise is added to the signal to be analyzed so that the signals of different time scales can be automatically separated to the corresponding reference scales. This method is mainly to add white noise to the signal to supply some missing scale which has good performance in signal decomposition.

In recent years, there are some time series prediction works using time series decomposition in several search areas. A regression model combined with wavelet transform is proposed to forecast the future value of the S&P 500 [11]. EMD is used for electricity load forecasting [12]. Besides, time series decomposition has been applied to disease prediction work. An ensemble model for chickenpox forecast utilizes the STL decomposition to generate the input of the model. Wavelet-ARIMA model got a good performance in COVID-19 case prediction [13]. An improved EEMD algorithm is used to decompose the diarrhea time series [14]. A TDDF model utilizes heterogeneous data to predict the HDMD cases [15].

The HFMD outpatient time series data is applied in our study. The spread of HFMD is easily affected by many external factors. Thus, the processing of the time series is difficult. But the adaptive nature of EMD overcomes this problem. In this paper, we introduce EMD to process our input data.

2.2. Fusion Methods. Time series forecasting has been a subject of interest in several different research areas including disease control and prevention. In the practical problems of nature, things are not isolated from each other but inextricably connected. The same goes for HFMD. Many studies have fused exogenous data to improve the accuracy of prediction.

The spread of HFMD is influenced by many external factors, such as meteorological factors including temperature, humidity, rapid climate change, local policies, air quality, and population [15–17]. Besides, making good use of some data can help researchers to predict, for instance, the search engine query data [18, 19].

Several methods using exogenous data are collected, and these models can be classified into two categories—stochastic methods and learning methods. Stochastic methods usually combine the past data and exogenous data by a linear method and then learn a linear function to get prediction results [16, 17, 19–23]. The main differences between these methods are the regression of target variables, functions on exogenous data, and the decomposition of exogenous data. In the past few years, exogenous data has been widely used in learning methods. These methods can be roughly divided into the following three categories:

(1) *Traditional Learning Methods Using Exogenous Inputs*. The most common models are multiple linear regression (MLR), support vector regression (SVR), and neural network. For these methods, exogenous data is treated as an input dimension, just like the past data, in which each element of the inputs is equally treated. To prevent data jitter, these methods need to be validated.

(2) The learning methods focus on temporal, which inputs of different categories are differential treatment, such as [24–26]. For these methods, the temporal dynamics of input data is captured to use RNN structures, and a nonlinear mapping from inputs to the target is learned from training data. To differently treat exogenous inputs and target inputs, the encoder-decoder structure is employed to do time series prediction tasks. The encoder-decoder framework consists of two RNN layers and maps input sequence to output sequence [27]

(3) *Temporal Attention Learning Methods*. The attention mechanism is fused into sequential models to predict future values, such as TPA-LSTM [28], DA-RNN [29], HRHN [30], and LSTNet [31]. These models have strong memory abilities in keeping numerous samples. Especially for small-scale infection data, the training loss value would be very small, but the accuracy would be worse than the general methods and is not general enough.

Though the exogenous data can help to improve accuracy, it still has some unavoidable defects. That is, the exogenous data requires a mass of energy to collect and organize and it is unavailable sometimes. Therefore, it is not always wise to do prediction relying on exogenous data, especially in real-time systems. It is almost impossible to integrate required data into the model dynamically. Considering the drawbacks of exogenous data, we discussed above, our attention focuses on target data itself and we do not utilize the exogenous data.

3. The Proposed CARD

This section formulates the problem and illustrates our approach.

Figure 1 shows an overview of the proposed model. The model consists of 3 stages: data preprocessing (left), Concurrent Autoregression with Decomposition (upper right), and data postprocessing (bottom right). For any module in Figure 1, if it makes any changes in the input data, then this module will be connected to the following modules using dotted lines.

In the data preprocessing stage, the input data is the HFMD outpatient. The outpatient data is normalized and then further segmented. Finally, they are decomposed into finite IMFs and residual by EMD. In the CARD, softmax function is introduced to avoid unfairness in feature extraction. Two linear autoregression components are used to mine the sequence feature details and enhance the feature representation of input data. At last, the output of two linear components is fused and another linear component is applied to generate the predicted value. In the data postprocessing stage, the final result is generated and evaluated after denormalization.

3.1. Problem Formulation and Notations. The main notations are explained in Table 1.

Windows size T. A window is a subsequence of the original data. T is the length of the subsequence. And the subsequence is the data in a certain interval be observed to predict the value of future time point.

IMF. If we do not have a termination, the EMD algorithm will loop an infinite number of times. In our experiments, we set a max number of the IMF which is symbolized as K to stop the decomposition.

The problem of this paper can be addressed as the problem of time series prediction missions. A time series is a list of continuous history observation values with equal time intervals. Our goal is to get a predicted value of the outpatient value of the next day.

It is a mapping from the history observation time series and IMFs to the future outpatient value. The symbol $y_t \in \mathbb{R}^1$ is the value at time t. The history observation values with window size T are symbolized as $[y_1, y_2, \cdots, y_T]$. And $\mathbf{D}(y_1, y_2, \cdots, y_T)$ is the matrix obtained by decomposing the windowed time series. $\hat{y}_{T+1}$ denotes the predicted value at time $T + 1$. The mapping process can be formulated as follows:

$$\hat{\mathbf{y}}_{T+1} = F(y_1, y_2, \cdots, y_T, \mathbf{D}(y_1, y_2, \cdots, y_T)). \tag{1}$$

In this study, $[y_1, y_2, \cdots, y_T]$ denotes the HFMD outpatient window size T.

3.2. Data Preprocessing. Normalization. The normalization operation scales the data in a specified range. In order to

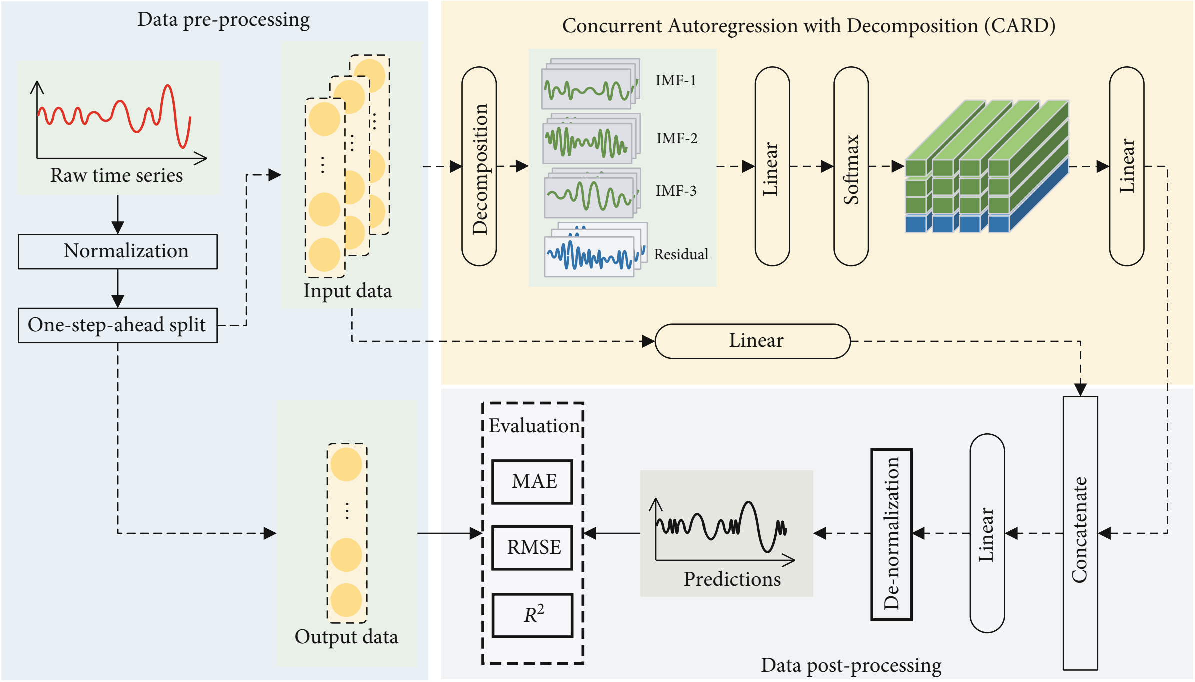


Figure 1: The scheme illustration of the proposed CARD.

Table 1: Notation and semantic.

Notation	Semantic
K	Number of points in the time series
T	Window size
k	Number of IMF
$\mathbf{O}$	Input matrix, $\mathbf{O} \in \mathbb{R}^{1\times K}$
$\mathbf{Y}$	Matrix after split, $\mathbf{Y} \in \mathbb{R}^{(K-T-1)\times T}$
$\mathbf{X}$	Matrix after decomposition, $\mathbf{X} \in \mathbb{R}^{(K-T-1)\times T\times k}$
$\hat{\mathbf{Y}}_{T+1}$	Output matrix, $\hat{\mathbf{Y}}_{T+1} \in \mathbb{R}^{1\times(K-T)}$

avoid large data dominance caused by the difference of data magnitude, normalization is essentially requisite.

Min-max normalization (0-1 normalization) is a widely use method in time series normalization. It is a linear transformation of the original data, making the result fall into the interval of (0,1). The original data can maintain the difference of value after the linear transformation. Thus, Min-max is suitable to normalize the outpatient time series in our study. The formula of the Min-max normalization is expressed as follows:

$$\mathbf{x}' = \frac{\mathbf{x} - \min(\mathbf{x})}{\max(\mathbf{x}) - \min(\mathbf{x})}, \tag{2}$$

where $\mathbf{x}$ denotes a sample of observed samples, $\mathbf{x}'$ is the normalization result, $\min(\mathbf{x})$ is the smallest value in the samples, and $\max(\mathbf{x})$ is the biggest.

Segmentation. The purpose of segmentation is to transform time series data into supervise data. For a given time series $\mathbf{Y}$ with K points, the segmentation formula is as follows:

$$\begin{bmatrix} y_1 & y_2 & \cdots & y_T \\ y_2 & y_3 & \ddots & y_{T+1} \\ \vdots & \ddots & \ddots & \vdots \\ y_{K-T-1} & \cdots & \cdots & y_{K-1} \end{bmatrix} \rightarrow \begin{bmatrix} y_{T+1} \\ y_{T+2} \\ \vdots \\ y_K \end{bmatrix}, \tag{3}$$

where the left matrix is the input data and the right matrix is the output data.

Empirical Mode Decomposition. In this paper, we use EMD [13] to do data decomposition. We perform time series decomposition on the supervision data generated by segmentation. For each sequence, we decompose it into 3 IMFs and a residual.

An IMF must satisfy the requirements as follows:

(1) In any local time scale, the number of extrema and the number of points cross zero must be equal or the difference is 1

(2) At any point, the mean value of the upper envelop defined by the local maxima and the lower envelope defined by local minima is close to 0

The procedures of EMD algorithm are shown in Algorithm 1.

Let $\mathbf{X} = [\mathrm{IMF}_1, \mathrm{IMF}_2, \cdots, \mathrm{IMF}_k, \mathrm{residual}]$ be the decomposed data.

```
Input: The original signal $x(t)$, max-IMF $k$
Output: $k$ IMFs and a residual
$i = 1$;
while $i < k$ do
$\mathrm{Upper}(t) = \text{spline local maxima } x(t)$;
$\mathrm{Lower}(t) = \text{spline local minima } x(t)$;
$\mathrm{Avg}(t) = 1/2(\mathrm{Upper}(t) + \mathrm{Lower}(t))$;
$h(t) = x(t) - \mathrm{Avg}(t)$;
if $h(t)$ meets two requirements of IMF do
$\mathrm{IMF}_i = h(t)$;
$i++$;
$x(t) = \mathrm{Avg}(t)$;
$\mathrm{Residual} = x(t)$
```

Algorithm 1: EMD

3.3. Concurrent Autoregression. The processing of IMF. We utilize the softmax function to process the IMF future. The softmax function is an extension of the logistic function. This function maps a k-dimensional vector containing any real number to another k-dimensional real-valued vector. Such that each element is in the interval (0, 1), and the sum of all elements is 1. After the process of softmax, the largest value is highlighted and the other components that are far below the maximum value are suppressed. The formula of softmax function is expressed as follows:

$$\boldsymbol{\omega}_i = \frac{e^{x_i}}{\sum_{j=1}^{k} e^{x_j}}, \tag{4}$$

where $\mathbf{x_i}$ is the output value of the i-th input vector. $\boldsymbol{\omega}$ is the weight matrix. k is the number of output elements. The generation of input memory $\mathbf{a}_i$ is based on the input vector $\mathbf{X}$ and the weight matrix $\boldsymbol{\omega}$. The formula is expressed as follows:

$$\mathbf{a}_i = \boldsymbol{\omega} \times \mathbf{X}. \tag{5}$$

CARD employs a linear layer to receive a regression result of IMF. The formula is expressed as follows:

$$\mathbf{e}_q = \sum \boldsymbol{\omega}_q \times \mathbf{a}_i + \mathbf{b}_q, \tag{6}$$

where $\mathbf{e}_q$ is the weighted IMF feature matrix, $\boldsymbol{\omega}_q$ is the weight corresponding to the input dimension, and $\mathbf{b}_q$ is a bias value.

The processing of the HFMD outpatient data. The processing of outpatient data is essentially the same as that of IMFs. The difference is the softmax function is not use for normalization. Only a linear component is applied to analyze the trends in outpatient data. The formula is expressed as follows:

$$\mathbf{e}_y = \sum \boldsymbol{\omega}_y \times \mathbf{y}_i + \mathbf{b}_y, \tag{7}$$

where $\mathbf{e}_y$ is the weighted outpatient feature matrix, $\boldsymbol{\omega}_y$ is the weight corresponding to the input dimension, and $\mathbf{b}_y$ is a bias value.

Concatenation. The CARD model combines the output of two concurrent working components by the cat function in PyTorch. The data is treated as the input for last linear module, and finally, this module generates a predicted value $\widehat{y}_{T+1}$. The generation of $\widehat{y}_{T+1}$ is formulated as follows:

$$\widehat{\mathbf{y}}_{T+1} = \Phi\left(\boldsymbol{\omega}\left[\mathbf{e}_{y;}\, \mathbf{e}_q\right] + \mathbf{b}\right), \tag{8}$$

where $[\mathbf{e}_{y;}\, \mathbf{e}_q]$ is the concatenated vector of dual side outputs, $\boldsymbol{\omega}$ is the weight of outputs from dual represented sources, $\widehat{\mathbf{y}}_{T+1}$ is the predicted value of the outpatient number in the next day, $\mathbf{b}$ is a bias value, and Φ represents the activation function.

3.4. Data Postprocessing. Denormalization. Denormalization is an inverse procedure of normalization, and the denormalization formula is applied to generate the final prediction results acquired from our model. The formula is expressed as follows:

$$\mathbf{x} = \mathbf{x}' \times (\max(\mathbf{x}) - \min(\mathbf{x})) + \min(\mathbf{x}). \tag{9}$$

The detail steps of the proposed CARD are shown in Algorithm 2.

4. Experimental Setup

This section configures our experiments. Section 4.1 introduces the dataset we use. Section 4.2 gives three evaluation metrics. And Section 4.3 presents the implementation of our model and the baseline models for comparison. All experiments are proceeding with the real-world HFMD outpatient case time series data.

4.1. Data. The real dataset we applied in our experiments is HFMD outpatient case data which is collected from the Xiamen Center for Disease Control and Prevention (XCDC). This dataset is the daily record data from January 1, 2012, to December 30, 2018. A total of 2555 sample points are included. In Figure 2, the time series is shown at one-year intervals.

4.2. Metric. To measure the performance of our proposed model and compare our model with the selected baseline models, 3 widely used standard methods are adopted in our experiments, and the formulas are defined as follows:

$$\mathrm{MAE} = \frac{1}{T}\sum_{t=1}^{T}(|y_t - \widehat{y}_t|), \tag{10}$$

$$\mathrm{RMSE} = \sqrt{\frac{1}{T}\sum_{t=1}^{T}(y_t - y\wedge_t)^2}, \tag{11}$$

$$R^2 = 1 - \frac{\sum_{t=1}^{T}(y_t - y\wedge_t)^2}{\sum_{t=1}^{T}(\bar{y}_t - y\wedge_t)^2}. \tag{12}$$

In these equations, the parameter y_t is the real

```
Input: Observed HFMD outpatient time series $\mathbf{O} \in \mathbb{R}^{1\times K}$, window size $T$, max-IMF $k$
Output: Prediction value for future cases $\widehat{\mathbf{Y}}_{T+1} \in \mathbb{R}^{1\times(K-T)}$
$\mathbf{O}'$ ← normalization $\mathbf{O}$ using Equation (2);
$\mathbf{Y}$ ← Split using Equation (3);
$\mathbf{X}$ ← Time series decomposition using Algorithm 1;
for each sample $\mathbf{y}$ in $\mathbf{Y}$ do
  for $i$ ← 1 do
    for $j$ ← 1 do
      $\mathbf{e}_y \leftarrow \mathbf{y}_{i,j}$ using Equation (7);
for each sample $\mathbf{x}$ in $\mathbf{X}$ do
  for $i$ ← 1 do
    for $j$ ← 1 do
      for $n$ ← 1 do
        $\boldsymbol{\omega}$ ← softmax $\mathbf{x}_{i,j}$ using Equation (4);
$\mathbf{A} \leftarrow \boldsymbol{\omega}$ and $\mathbf{X}$ using Equation (5);
$\mathbf{e}_x \leftarrow \mathbf{A}$ using Equation (6);
$\widehat{\mathbf{y}'}_{T+1} \leftarrow \mathbf{e}_x$ and $\mathbf{e}_y$ using Equation (8);
$\widehat{\mathbf{y}}_{T+1}$ ← denormalization $\widehat{\mathbf{y}'}_{T+1}$ using Equation (9);
```

Algorithm 2: CARD

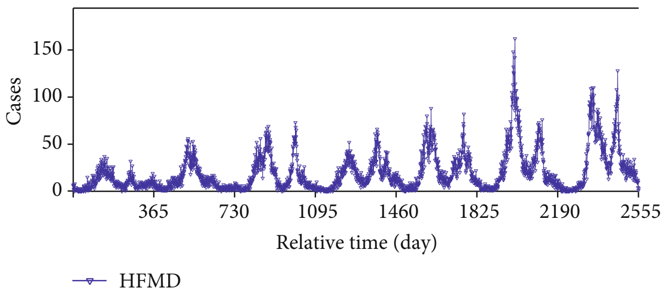


Figure 2: The distribution of outpatient cases ranges from Jan 1, 2012, to Dec 30, 2018.

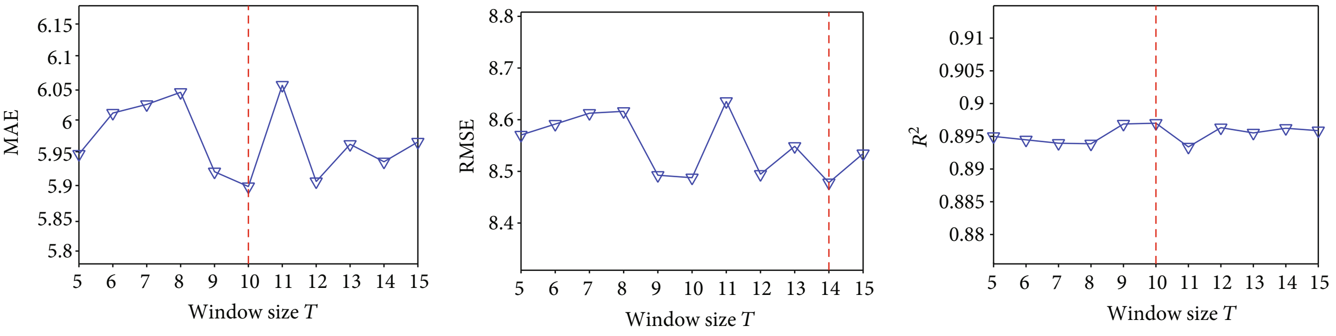
observation value at time t, and $\widehat{y}_t$ is the predicted value predicted by model at the same time.

MAE is a basic and universal metric in regression mission. Compared with MAE, RMSE has the same degree as the data. For R^2, the denominator is understood as the dispersion degree of the original data, and the molecule is the error between the predicted data and the original data. The division of the two can eliminate the influence of the dispersion degree of the original data. These three metrics can be used together to evaluate the performance of the model comprehensively and objectively.

4.3. Configuration. Parameter settings. In our experiments, the target data is divided into two parts: training set (80%) and test set (20%). The batch size is set to 32. A set of experiments are completed to find the best values of window size, and the results are shown in Figure 3, and as we can see, the best performance is achieved when $T = 10$. For each experiment, we chose the learning rate between 0.0005 and 0.002 for a step 0.0005 to acquire the best performance of every model. We repeat each set of experiments five times and take the average value to obtain the final result. Thus, the result is stable and has a high level of credibility.

Decomposition algorithm. RobustSTL is more suitable for long-time series processing, and the time series we used is too short for this method. Therefore, we only consider wavelet transform, EMD, and EEMD as the time series decomposition method candidates. There are four experiments for comparison that have been done, and the results are shown in Figure 4. "IMF3" means the original data will be decomposition into three IMFs and one residual. Both "db" and "sym" are commonly used wavelet basis functions. "db" is the abbreviation of Daubechies, and "db2" represents a wavelet of order 2. "sym" is symlets and "sym3" means a wavelet of order 3. As we have seen in Figure 4, the wavelet transform-based approach has a time advantage, while the EEMD algorithm consumes too much computational time. Although the EMD algorithm is at a time disadvantage, it takes the lead in three metrics. Thus, this configure is applied in the formal experiments.

Baseline models. To verify the effectiveness of the EMD function and local fusion, experiments based on multiple models are necessary. MLR [32], LSTM [33], GRU [34], ED

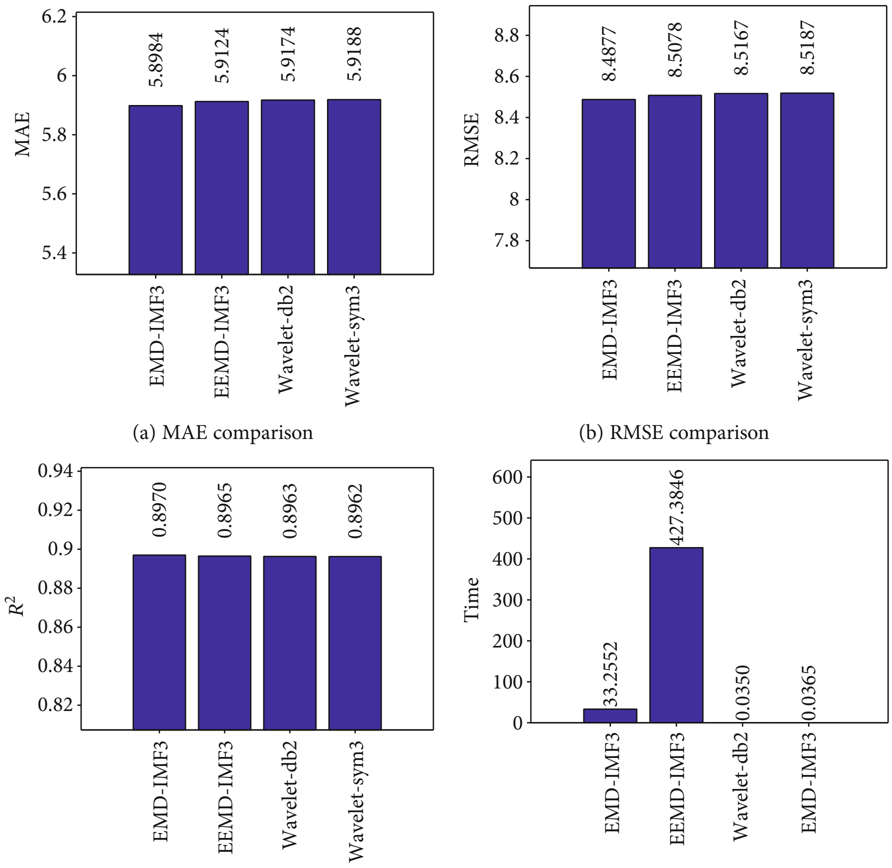


(a) MAE comparison

(b) RMSE comparison

(c) R^2 comparison

Figure 3: The comparison of CARD performance at different values of T in terms of MAE, RMSE, and R^2.



(a) MAE comparison

(b) RMSE comparison

(c) R^2 comparison

(d) Computation time comparison

Figure 4: The comparison of four methods in terms of MAE, RMSE, R^2, and computation time.

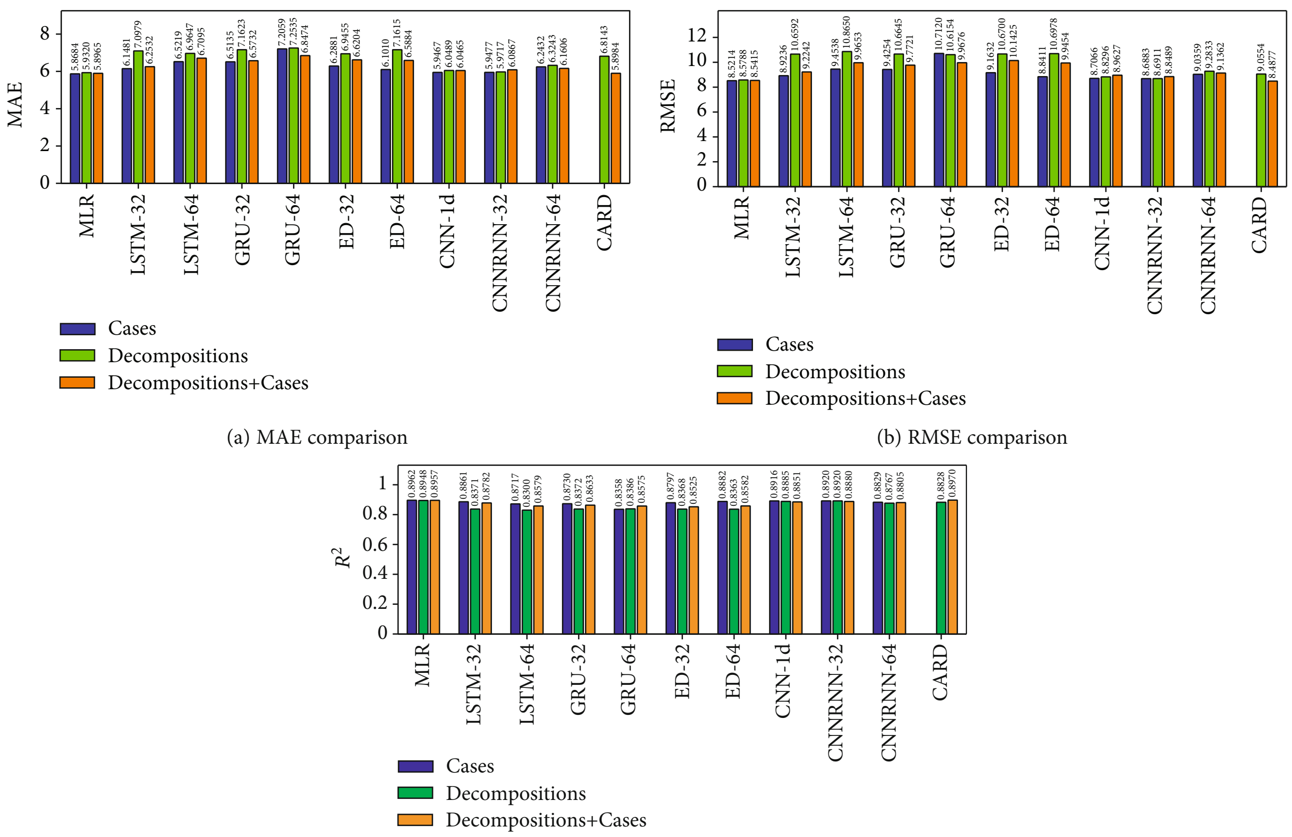
[35], CNN-1d, and CNN-RNN [36] are selected as the baseline models. To explain, MLR is a widely used regression model in many research areas. LSTM, GRU, and ED are improved neural networks based on RNN. CNN-RNN is a hybrid model of CNN and RNN.

Experiment process. All experiments could be divided into 3 groups. The main difference between the three groups of experiments is their input data. The input data of the first group of experiments contains only historical data, and the second group uses only the data after time series decomposition. The last group of experiments takes both historical and decomposed data as input. As a result, the final group has the best performance. Details are discussed in the next section.

5. Results and Analysis

This section gives prediction results, comparisons, and analyses.

5.1. Effects on Decomposition and Fusion. In this subsection, we investigate the effects of decomposition and fusion. As we can see in Figure 5, the result of three metrics shows that



(a) MAE comparison

(b) RMSE comparison

(c) R^2comparison

FIGURE 5: The comparison of eleven methods on three groups of inputs in terms of MAE, RMSE, and R^2. The windows size T is fixed at 10.

the CARD model we proposed has the ideal performance compared with baseline models.

Time series decomposition is an important part of this study. We decompose the HFMD outpatient time series into finite and multitime scale IMFs and a residual; then, each subsequence is modeled and predicted with a local linear predictor separately. The single IMF contains a specific physical meaning, such as seasonality and trend. Each sequence is treated equally in the model. Compared with the original data, each IMF can represent the local features by itself. This means that separate predictions for each sequence and then fusion may give better results than using only the raw data, and the experimental results proved this.

The prediction accuracy of all baseline models has increased after fusing the HFMD outpatient case data. This result shows the superiority of data fusion. A possible explanation is that existing models do not work well with complex time series like IMF, and some methods cannot capture the relation between different sequences. The data processing of the CARD can be divided into two stages. In the first stage, each sequence is predicted separately, and then, the results are fused. In the second stage, the predicted values are obtained from the fused data, which can analyze the relationship between each sequence. So, we get better results than other models. By the way, the IMFs may lose some features in the original data. These defects are more obvious with the short and complex time series data. However, the fusion of IMF and case data overcomes this shortage. That may explain why all models have various degrees of improvement after fusion.

5.2. Comparison. The main results are shown in Figure 5. The major results can be observed, and the analysis of them is as follows:

Out of all the models, CARD performs the best. In detail, MLR is the second-best model. Compared to MLR, CARD is slightly behind in MAE and RMSE, and we are slightly ahead in R^2. In addition, we are at least 0.1, 0.4, and 0.1 ahead of the other models in three metrics. The advantages of our model are described in Section 5.1, and these should explain the leading position of our model.

In the experiments using only decomposed data as input, several baseline models showed various degrees of degradation in performance. And their performance is improved if the outpatient data is added to the dataset. However, as we can see in Figure 5, the best performance of these models is still obtained in the first set of experiments—the input is outpatient data. In contrast, the performance of CNN-1d and MLR shows only small fluctuations. One possible explanation is that the EMD algorithm filters peaks in the time series while CNN-1d and MLR are insensitive to peaks. Therefore, the accuracy of these two models is not affected much. LSTM, ED, and GRU study the dependence of time series, and since these models cannot capture the relationship between the series, the IMF may negatively affect the prediction accuracy. CARD performs weighting at each time point and predicts each IMF separately. Finally, the model generates a result by fusion. Thus, CARD solves these problems and obtains better performance.

Although the CARD model does not make revolutionary advances, however, the model is much less computationally intensive compared to most neural network models. Therefore, the model has relatively low hardware requirements. Moreover, this model still has good predictive performance when using only historical data, which means that the data needed to run the model is easily available. This further lowers the threshold for practical using the model. Therefore, our proposed model has good prospects for practical applications.

6. Conclusions

Our experiment indicates that data decomposition and local fusion can improve prediction performance. In this paper, we propose a time series decomposition and local fusion model named CARD for HFMD outpatient case prediction. The main conclusions of this study are shown as follows:

(1) Compared with wavelet transform and EEMD, the EMD method has advantages in predicting accuracy in terms of HFMD outpatient prediction. Therefore, EMD is suitable for HFMD outpatient time series

(2) The fusion model we proposed is superior to the most general methods, which means that such a model still has great potential in infectious disease forecasting

Our study must go further research. In this paper, we do not test the predicting accuracy on the multistep prediction. In the next step, we can try to extend our model to multistep times series prediction and other diseases.

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 partly supported by Jimei University (nos. ZP2021013 and ZP2020043), the Science Project of Xiamen City (CN) (no. 3502Z20193048), the Education Department of Fujian Province (CN) (nos. JAT200277 and JAT200232), and the Natural Science Foundation of Fujian Province (CN) (no. 2019J01713). We gratefully acknowledge Xiamen Center for Disease Control and Prevention (XCDC) for sharing the data. We gratefully acknowledge the editor and anonymous reviewers for their valuable insights and suggestions which enormously benefited the paper.

References

[1] C C for Disease Control, Prevention, “The general situation of notifiable infectious diseases in china in 2019,” 2020, 2020, http://www.nhc.gov.cn/jkj/s3578/202004/b1519e1bc1a944fc8ec176db600f68d1.shtml.

[2] L. Liu, R. Luan, F. Yin, X. Zhu, and Q. Lu, “Predicting the incidence of hand, foot and mouth disease in Sichuan province, China using the ARIMA model,” *Epidemiology and Infection*, vol. 144, pp. 144–151, 2016.

[3] Z. Du, L. Xu, W. Zhang, D. Zhang, S. Yu, and Y. Hao, “Predicting the hand, foot, and mouth disease incidence using search engine query data and climate variables: an ecological study in Guangdong, China,” *BMJ Open*, vol. 7, no. 10, article e016263, 2017.

[4] J. Gu, L. Liang, H. Song et al., “A method for hand-foot-mouth disease prediction using GeoDetector and LSTM model in Guangxi, China,” *Scientific Reports*, vol. 9, no. 1, p. 17928, 2019.

[5] Y. Wang, Z. Cao, D. Zeng, X. Wang, and Q. Wang, “Using deep learning to predict the hand-foot-and-mouth disease of enterovirus A71 subtype in Beijing from 2011 to 2018,” *Scientific Reports*, vol. 10, no. 1, p. 12201, 2020.

[6] A. Grossmann and J. Morlet, “Decomposition of hardy functions into square integrable wavelets of constant shape,” *SIAM Journal on Mathematical Analysis*, vol. 15, no. 4, pp. 723–736, 1984.

[7] Q. Wen, J. Gao, X. Song, L. Sun, H. Xu, and S. Zhu, “RobustSTL: a robust seasonal-trend decomposition algorithm for long time series,” in *Proceedings of the 33rd International Conference on Artificial Intelligence, AAAI Press*, pp. 5409–5416, Honolulu, Hawaii, USA, 2019.

[8] C. Robert, C. William, and T. Irma, “STL: a seasonal-trend decomposition procedure based on loess,” *Journal of Official Statistics*, vol. 6, no. 1, pp. 3–73, 1990.

[9] N. E. Huang, Z. Shen, S. R. Long et al., “The empirical mode decomposition and the Hilbert spectrum for nonlinear and non-stationary time series analysis,” *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, vol. 454, no. 1971, pp. 903–995, 1998.

[10] Z. Wu and N. E. Huang, “Ensemble empirical mode decomposition: a noise-assisted data analysis method,” *Advances in Adaptive Data Analysis*, vol. 1, no. 1, pp. 1–41, 2009.

[11] K. Zhang, R. Gencay, and M. Ege Yazgan, “Application of wavelet decomposition in time-series forecasting,” *Economics Letters*, vol. 158, pp. 41–46, 2017.

[12] Y. Yaslan and B. Bican, “Empirical mode decomposition based denoising method with support vector regression for time series prediction: a case study for electricity load forecasting,” *Measurement*, vol. 103, pp. 52–61, 2017.

[13] S. Singh, K. S. Parmar, J. Kumar, and S. J. S. Makkhan, “Development of new hybrid model of discrete wavelet decomposition and autoregressive integrated moving average (ARIMA) models in application to one month forecast the casualties cases of COVID-19,” *Chaos, Solitons & Fractals*, vol. 135, p. 109866, 2020.

[14] Y. Wang, J. Gu, Z. Zhou, and Z. Wang, “Diarrhoea outpatient visits prediction based on time series decomposition and multi-local predictor fusion,” *Knowledge-Based Systems*, vol. 88, pp. 12–23, 2015.

[15] Z. Wang, Y. Huang, B. He, T. Luo, Y. Wang, and Y. Lin, “TDDF: HFMD outpatients prediction based on time series

decomposition and heterogenous data fusion in Xiamen, China," in *Proceedings of the 15th International Conference Advanced Data Mining and Applications*, pp. 658–667, Dalian, China, 2019.

[16] J. Wang, Y. Xiao, and R. A. Cheke, "Modelling the effects of contaminated environments on HFMD infections in mainland China," *Biosystems*, vol. 140, pp. 1–7, 2016.

[17] Z. Yang, J. Hao, S. Huang et al., "Acute effects of air pollution on the incidence of hand, foot, and mouth disease in Wuhan, China," *Atmospheric Environment*, vol. 225, p. 117358, 2020.

[18] P. Wang, W. B. Goggins, and E. Y. Y. Chan, "Hand, foot and mouth disease in Hong Kong: a time-series analysis on its relationship with weather," *PLoS One*, vol. 11, no. 8, article e0161006, 2016.

[19] D.-C. Huang and J.-F. Wang, "Monitoring hand, foot and mouth disease by combining search engine query data and meteorological factors," *Science of the Total Environment*, vol. 612, pp. 1293–1299, 2018.

[20] S. Chen, X. Liu, Y. Wu et al., "The application of meteorological data and search index data in improving the prediction of HFMD: a study of two cities in Guangdong Province, China," *Science of The Total Environment*, vol. 652, pp. 1013–1021, 2019.

[21] C. Song, X. Shi, Y. Bo, J. Wang, Y. Wang, and D. Huang, "Exploring spatiotemporal nonstationary effects of climate factors on hand, foot, and mouth disease using Bayesian spatiotemporally varying coefficients (STVC) model in Sichuan, China," *Science of the Total Environment*, vol. 648, pp. 550–560, 2019.

[22] Y. Yang, E. You, J. Wu et al., "Effects of relative humidity on childhood hand, foot, and mouth disease reinfection in Hefei, China," *Science of the Total Environment*, vol. 630, pp. 820–826, 2018.

[23] Z. Du, W. R. Lawrence, W. Zhang, D. Zhang, S. Yu, and Y. Hao, "Interactions between climate factors and air pollution on daily HFMD cases: a time series study in Guangdong, China," *Science of The Total Environment*, vol. 656, pp. 1358–1364, 2019.

[24] Z. Wang, Y. Huang, B. He et al., "Short-term infectious diarrhea prediction using weather and search data in Xiamen, China," *Scientific Programming*, vol. 2020, Article ID 8814222, 12 pages, 2020.

[25] Z. Wang, Y. Huang, B. Cai, R. Ma, and Z. Wang, "Stock turnover prediction using search engine data," *Journal of Circuits, Systems and Computers*, vol. 30, no. 7, 2020.

[26] Y. Wang, J. Li, J. Gu, Z. Zhou, and Z. Wang, "Artificial neural networks for infectious diarrhea prediction using meteorological factors in Shanghai (China)," *Applied Soft Computing*, vol. 35, pp. 280–290, 2015.

[27] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proceedings of 28th International Conference on Neural Information Processing Systems, Curran Associates, Inc. Palais des Congrès de Montréal*, pp. 3104–3112, Montréal Canada, 2014.

[28] S. Shih, F. Sun, and H. Lee, "Temporal pattern attention for multivariate time series forecasting," *Machine Learning*, vol. 108, no. 8-9, pp. 1421–1441, 2019.

[29] Y. Qin, D. Song, H. Chen, W. Cheng, G. Jiang, and G. W. Cottrell, "A dual-stage attention-based recurrent neural network for time series prediction," in *Proceedings of the 26th International Joint Conference on Artificial Intelligence, International Joint Conferences on Artificial Intelligence*, pp. 2627–2633, Melbourne, Australia, 2017.

[30] Y. Tao, L. Ma, W. Zhang, J. Liu, W. Liu, and Q. Du, "Hierarchical attention-based recurrent highway networks for time series prediction," https://arxiv.org/abs/1806.00685.

[31] G. Lai, W. Chang, Y. Yang, and H. Liu, "Modeling long- and short-term temporal patterns with deep neural networks," in *Proceedings of the 41st International Conference on Research Development in Information Retrieval, ACM*, pp. 95–104, Ann Arbor, MI, USA, 2018.

[32] P. Wang and M. L. Puterman, "Mixed logistic regression models," *Journal of Agricultural, Biological, and Environmental Statistics*, vol. 3, no. 2, pp. 175–200, 1998.

[33] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[34] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," https://arxiv.org/abs/1412.3555.

[35] K. Cho, B. van Merrienboer, C. Gulcehre et al., "Learning phrase representations using RNN encoder-decoder for statistical machine translation," https://arxiv.org/abs/1406.1078.

[36] Y. Wu, Y. Yang, H. Nishiura, and M. Saitoh, "Deep learning for epidemiological predictions," in *The 41st International ACM SIGIR Conference on Research and Development in Information Retrieval, Association for Computing Machinery*, pp. 1085–1088, New York, NY, USA, 2018.

Hindawi
Wireless Communications and Mobile Computing
Volume 2021, Article ID 9922017, 13 pages
https://doi.org/10.1155/2021/9922017

Research Article

Separating Chinese Character from Noisy Background Using GAN

Bin Huang,[1] **Jiaqi Lin**,[1] **Jinming Liu**,[1] **Jie Chen**,[1] **Jiemin Zhang**,[1] **Yendo Hu**,[1] **Erkang Chen**,[1] **and Jingwen Yan**[2]

[1]*Computing Engineering College, Jimei University, Xiamen 361021, China*
[2]*College of Engineering, Shantou University, Shantou 515063, China*

Correspondence should be addressed to Erkang Chen; ekchen@jmu.edu.cn

Received 17 March 2021; Revised 7 April 2021; Accepted 20 April 2021; Published 3 May 2021

Academic Editor: Philippe Fournier-Viger

Separating printed or handwritten characters from a noisy background is valuable for many applications including test paper autoscoring. The complex structure of Chinese characters makes it difficult to obtain the goal because of easy loss of fine details and overall structure in reconstructed characters. This paper proposes a method for separating Chinese characters based on generative adversarial network (GAN). We used ESRGAN as the basic network structure and applied dilated convolution and a novel loss function that improve the quality of reconstructed characters. Four popular Chinese fonts (Hei, Song, Kai, and Imitation Song) on real data collection were tested, and the proposed design was compared with other semantic segmentation approaches. The experimental results showed that the proposed method effectively separates Chinese characters from noisy background. In particular, our methods achieve better results in terms of Intersection over Union (IoU) and optical character recognition (OCR) accuracy.

1. Introduction

Converting paper documents into electronic documents and then recognizing them using optical character recognition (OCR) technology have been widely used in daily life. In recent years, with the development of machine learning technology, the recognition accuracy of OCR has been greatly improved [1–3]. We can now process a document with both machine-printed text and handwritten text and then recognize them separately [4, 5]. Similar applications can be found in the archiving and processing of historical documents [6, 7]. In the field of education, related technologies for examination paper autoscoring have emerged, which greatly reduce burden for teachers and students. Taking Figure 1 as an example, an examination paper with students' answers can first be processed by OCR, and then the recognized answers can be evaluated and scored automatically by the machine. Under certain circumstance, since the test paper template cannot be easily obtained, it is also necessary to directly identify the printed test paper template.

In order to achieve examination paper autoscoring, one of the technical challenges to be solved is handling overlapping characters. This may happen when an elementary school student did not master writing well or put annotation on the test paper. The current OCR technology cannot handle the mixed situation of printed text and handwritten text in the same image. Generally, only a single type of text can be recognized by OCR technology [8]. Our early experiments showed that when recognizing printed text, the OCR accuracy was greatly reduced if there were handwritten strokes or handwritten characters around the printed text. Even worse was that the machine was not able to find the text area needed to be recognized. Therefore, it is desirable to separate the handwritten characters from the printed characters on the examination paper and then process different text types accordingly. Furthermore, for Chinese characters, the separation of handwriting and printing becomes more difficult because the font structure is far more complicated than Western fonts [9, 10]. A slight loss or increase of strokes may change the meaning of the characters completely, which

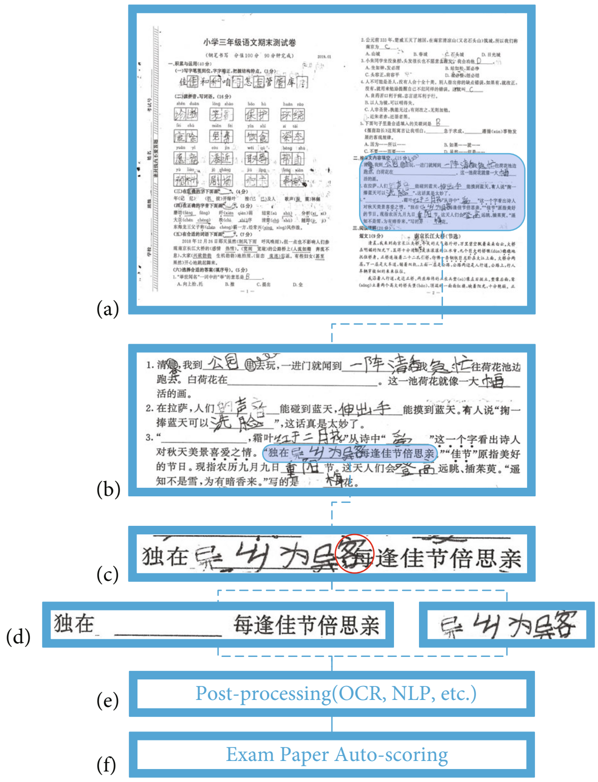


Figure 1: Basic process of examination paper autoscoring. (a) A sample examination paper which consists of both handwriting and printed text. (b) A subquestion with answers to be scored. (c) Handwriting touches or even overlaps with printed text. The red circle shows an example of overlapping characters. (d) The proposed method targets separation of overlapping Chinese characters into printed text (left rectangle) and handwriting (right rectangle). (e, f) After successful separation is made, postprocessing and autoscoring become feasible.

makes it difficult to separate effectively when handwriting fonts and printed fonts are highly overlapped.

Separating Chinese characters from noisy background (particularly with overlappings) can be considered an image semantic segmentation problem. Previous deep learning methods [11–13] have shown success in other applications. However, these methods have poor performance due to the complex structure of Chinese characters. To distinguish Chinese characters from similar fonts, we adopted a GAN-based approach [14–19]. A network, called DESRGAN, was developed to denoise background and reconstruct both the stroke structure and fine details of targeted Chinese characters. Our method used ESRGAN [19] as the basic network structure and applied dilated convolution to residual-in-residual dense blocks. A new loss function that can measure the integrity of the font skeleton was imposed. Then, the generator of the trained GAN model was used to separate targeted characters.

Our main contributions include the following: (a) we proposed a new network structure and a loss function that achieves the goal of Chinese character separation from noisy background, especially when characters are highly overlapped; (b) the proposed method achieved the best results in both IoU and OCR accuracy; and (c) our dataset (upon request) for further research is provided.

2. Related Work

Many applications in document processing need to solve the problem of separation of handwriting and printed. The Maurdor project created a realistic corpus of annotated documents in French, English, and Arabic to support the efficient development and evaluation of extraction method [20]. DeepErase [21] uses neural networks to erase ink artifacts on scanned documents and only extract text written by a user. The ink artifacts that DeepErase targets mainly include a tabular structure, fill-in-the-blank boxes, and underlines. Guo and Ma [22] used a machine-printed and handwritten annotation discrimination algorithm based on the Hidden Markov Model. Solely focusing on English and other Latin languages, their algorithm can locate the position of the handwritten part in the document in the form of a bounding box. Zagoris et al. [23] proposed a method of recognizing and separating handwritten content from document images mixed with handwritten and printed characters through the bag of visual word model. Their method first computes a descriptor for each block of interest and then classifies the descriptor into handwritten text, machine printed text or noise. However, few research has been focusing on highly overlapped texts, especially Chinese characters that are structurally more complex than English or other Latin languages.

Recent deep learning methods provide new ways for solving the separation of handwriting and printed. Li et al. [5] handles printed/handwritten text separation within a single framework by using conditional random fields. Their algorithm only performs extraction at connected component (CC) level. Each CC is classified into printed and handwritten no matter it is overlapping or not. U-Net [11], which performs well in many segmentation tasks, builds upon only convolution layers and the idea of propagating context information to higher resolution layers during upsampling. Pix2Pix [17] translates an input image into a corresponding output image. With a paired training dataset, it can output sharp and realistic images. Such features make it attractive for solving our character segregation problem. However, a paired training dataset may not be easy to find in real-world applications. CycleGAN [16] is an approach for learning to translate an image from a source domain to a target domain without paired examples. CycleGAN's coding scheme is to hide part of the information about the input image in low-amplitude, high-frequency signal added to the output image [14]. Another way to solve the separation of printed is to treat the image overlapped by handwriting and printed as a low-resolution picture, and the neural network determines which part needs to be enhanced in the process of single-image super-resolution. SRGAN [18] takes advantage of a perceptual loss function which consists of an adversarial loss and a content loss. Based on SRGAN, ESRGAN [19] improves the network structure by introducing the residual-in-residual dense block and computes perceptual

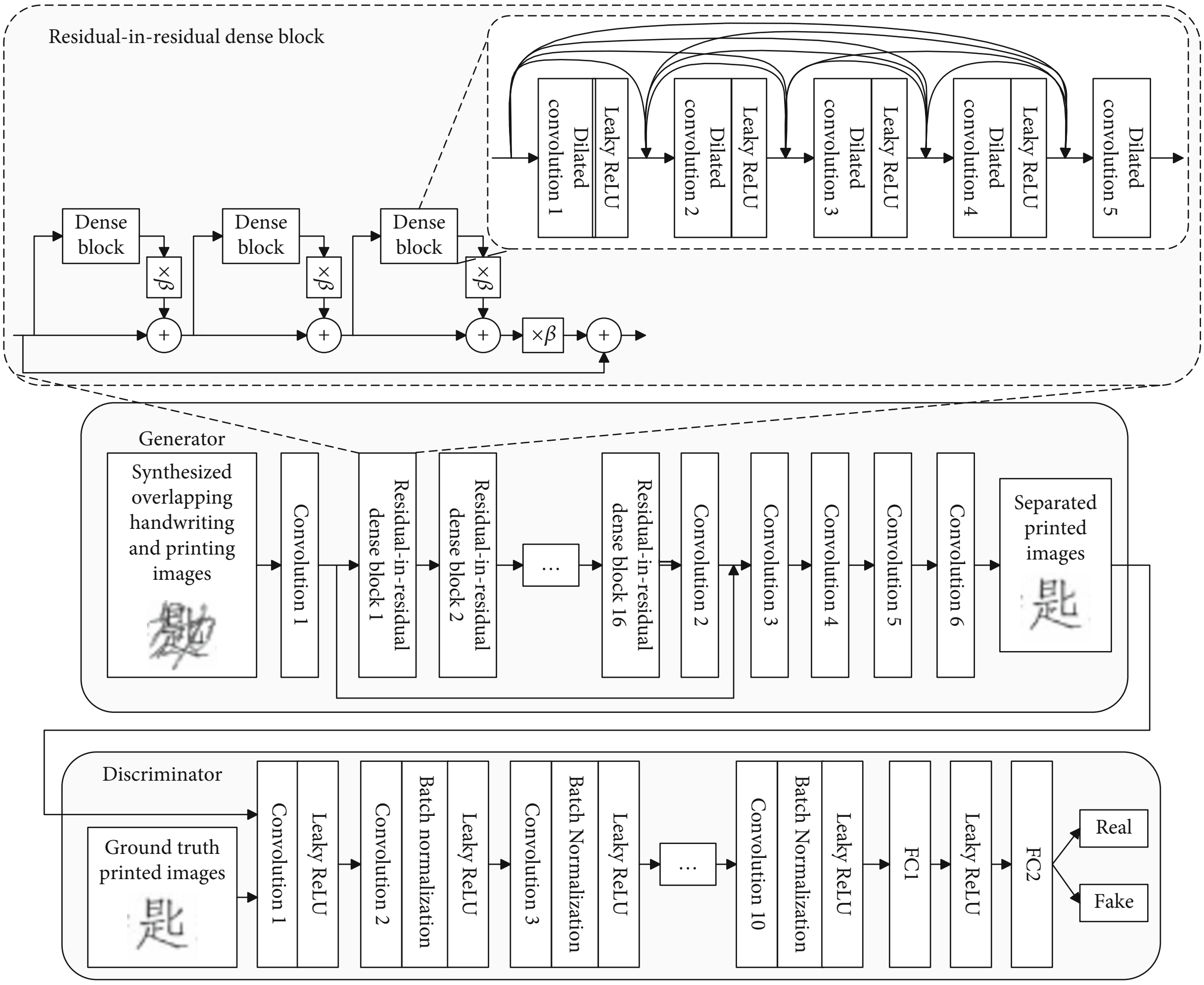


Figure 2: The network structure of the DESRGAN. An image with overlapping handwritten and printed characters is first processed by a series of convolution and RRDB modules. There is an operation of dilated convolution inside each residual-in-residual dense block. The generator outputs separated printed part or handwritten part from the overlapping. The discriminator classifies separated printed or handwritten characters and ground truth into real or fake.

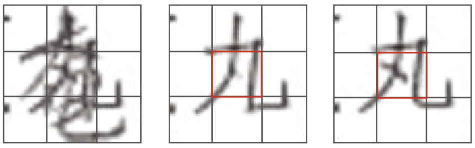


Figure 3: Image gridding (3×3 or 5×5) in the calculation of the integrity loss. From left to right: the overlapping image I^{OL}, the recovered image $G(I^{\mathrm{OL}})$, and its ground truth I^{GT}. Note that our integrity loss will focus on the center cells of $G(I^{\mathrm{OL}})$ and I^{GT} which are severely inconsistent.

loss by using features before activation instead of after activation. These techniques significantly improve the overall visual quality of reconstruction. Due to its versatility, GAN-based super-resolution techniques can potentially improve poor quality of document images, which is attributed to low scanning quality and resolution. Lat and Jawahar [24] super-resolve the low resolution document images before passing them to the OCR engine and greatly improve OCR accuracy on test images. However, we found that existing approaches could not provide satisfactory segregation results.

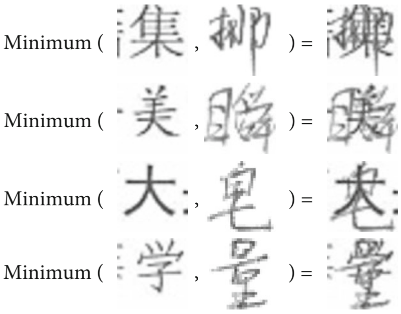


Figure 4: A simple example of image synthesis of overlapping of handwritten and printed characters. The three columns from left to right are printed Chinese, Chinese handwriting, and synthesized overlapping character.

Besides, there are research efforts toward handwriting synthesis. Graves [25] utilizes Long Short-term Memory recurrent neural networks to generate highly realistic cursive

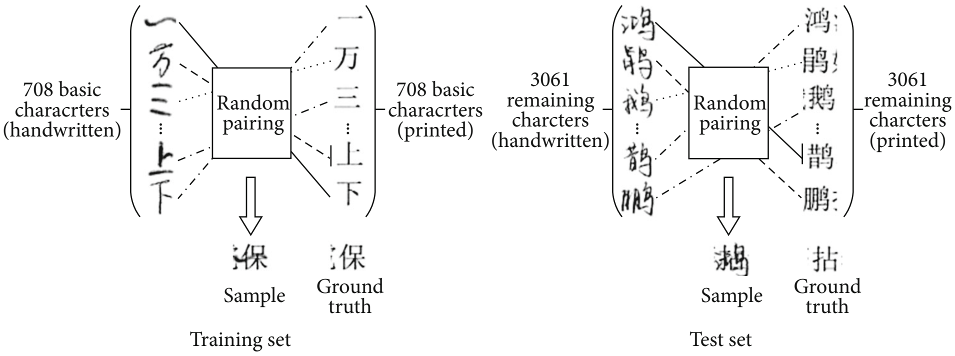


Figure 5: Data synthesis method. For each font type, 708 basic Chinese characters which contain various basic components are chosen to synthesize overlapping scenarios. Handwritten characters and printed characters are then randomly paired and overlapped to constitute data samples in training set. Synthesized in a similar way, the test set contains more Chinese characters and the character structure is more complex. Each data sample has a corresponding ground truth (i.e., original printed or handwritten character) for evaluating performance. The total size of the test set reaches approximately 12,200 unique overlapping characters.

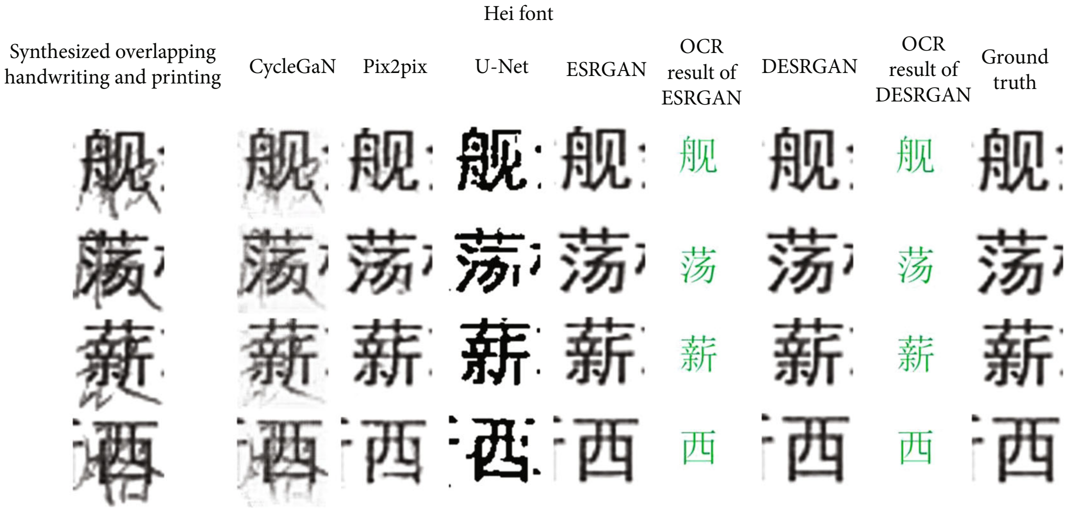


Figure 6: Results of separating printed Chinese characters in Hei font. Columns from left to right: synthesized handwritten/printed overlapped data sample in test set and results of separation of printed characters by CycleGAN, Pix2pix, U-Net, ESRGAN, and DESRGAN. The ground truth and OCR recognition results are also provided. U-Net classifies each pixel into one of two categories and generates a binary image.

handwriting in a wide variety of styles. His algorithm employs an augmentation that allows the network to generate data sequences conditioned on some high-level annotation sequence (e.g., a character string). Lian et al. [10] propose a system to automatically synthesize personal handwriting for all (e.g., Chinese) characters in the font library. Their work showed feasibility of learning style from a small number (as few as 1%) of carefully selected samples handwritten by an ordinary person. Although the handwriting fonts produced by their models have better visual effects, their offline processing flow requires the preparation of the writing trajectory of each stroke for all characters, which requires a lot of manual effort. Zhang et al. [9] use a recurrent neural network as a generative model for drawing Chinese characters. Under their framework, a conditional generative model with character embedding is employed to indicate the RNN of the identity of the character to be generated. The character embedding, which is jointly trained with the generative model, essentially limits the model to search the characters with similar writing trajectory (or similar shape) in the embedded space. Chang et al. [26] formulate the Chinese handwritten character generation as a style learning problem. Technically, they use CycleGAN to learn a mapping from an existing printed font to a personalized handwritten style. Our work referred to these methods to construct a dataset for training and evaluating the proposed DESRGAN.

3. Design

Based on the GAN architecture, our method is shown in Figure 2. Given a Chinese character with noisy background (e.g., overlapping), the generative network separates the targeted character, which can be printed or handwritten, from the input image. Since each stroke in a Chinese character is almost indispensable, extra attention should be paid to maintaining the integrity of the Chinese character structure. We

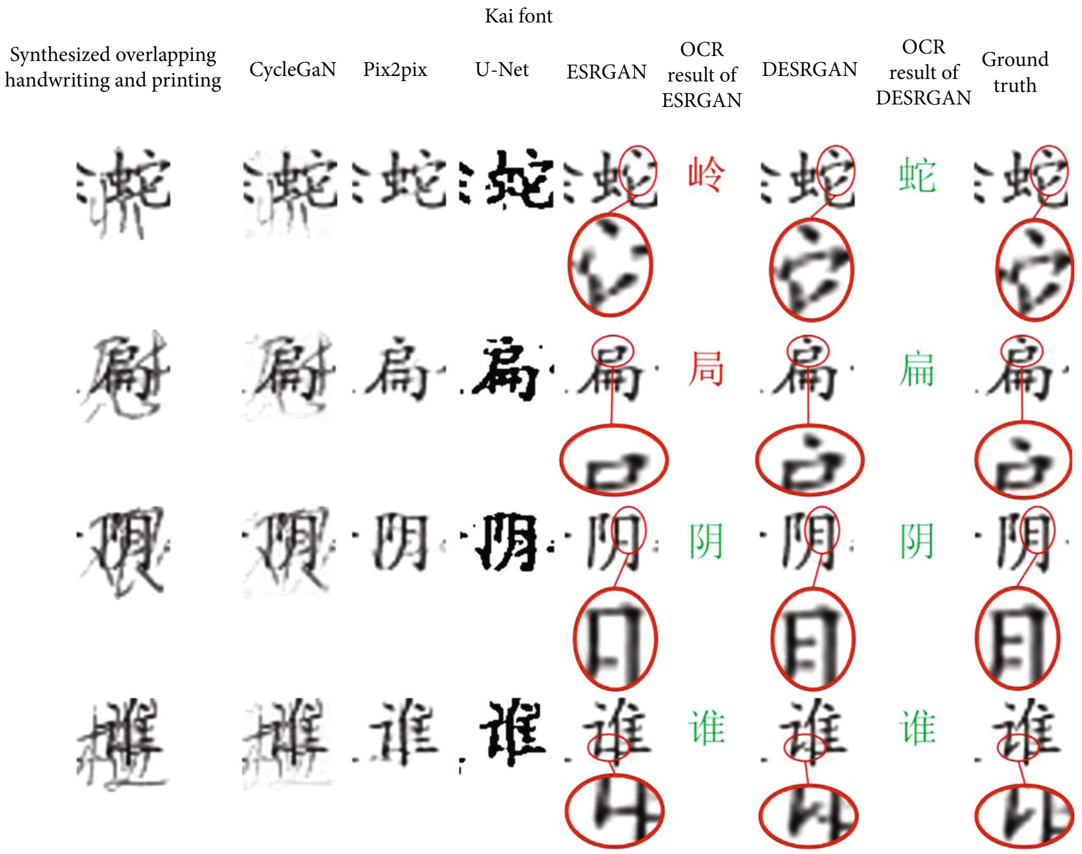


Figure 7: Results of separating printed Chinese characters in Kai font. Critical stokes or fine details of Chinese characters are clipped and enlarged for further examination. Wrong recognition results by OCR tool are colored in red while the correct ones in green.

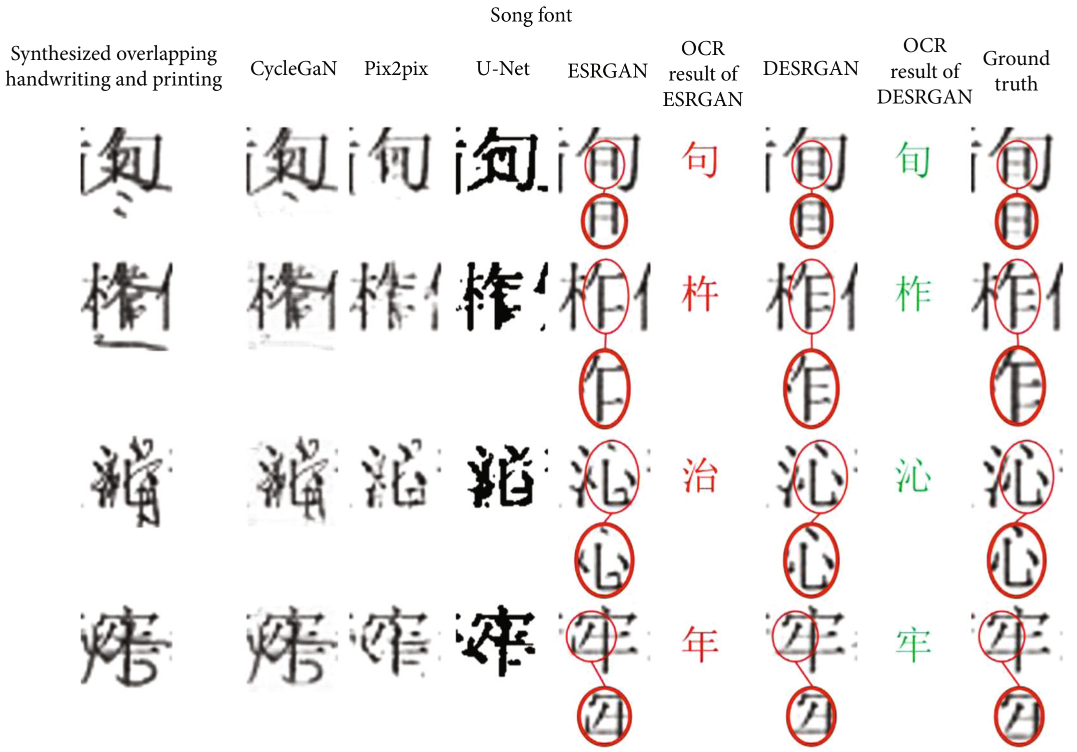


Figure 8: Results of separating printed Chinese characters in Song font. Critical stokes or fine details of Chinese characters are clipped and enlarged for further examination. Wrong recognition results by OCR tool are colored in red while the correct ones in green.

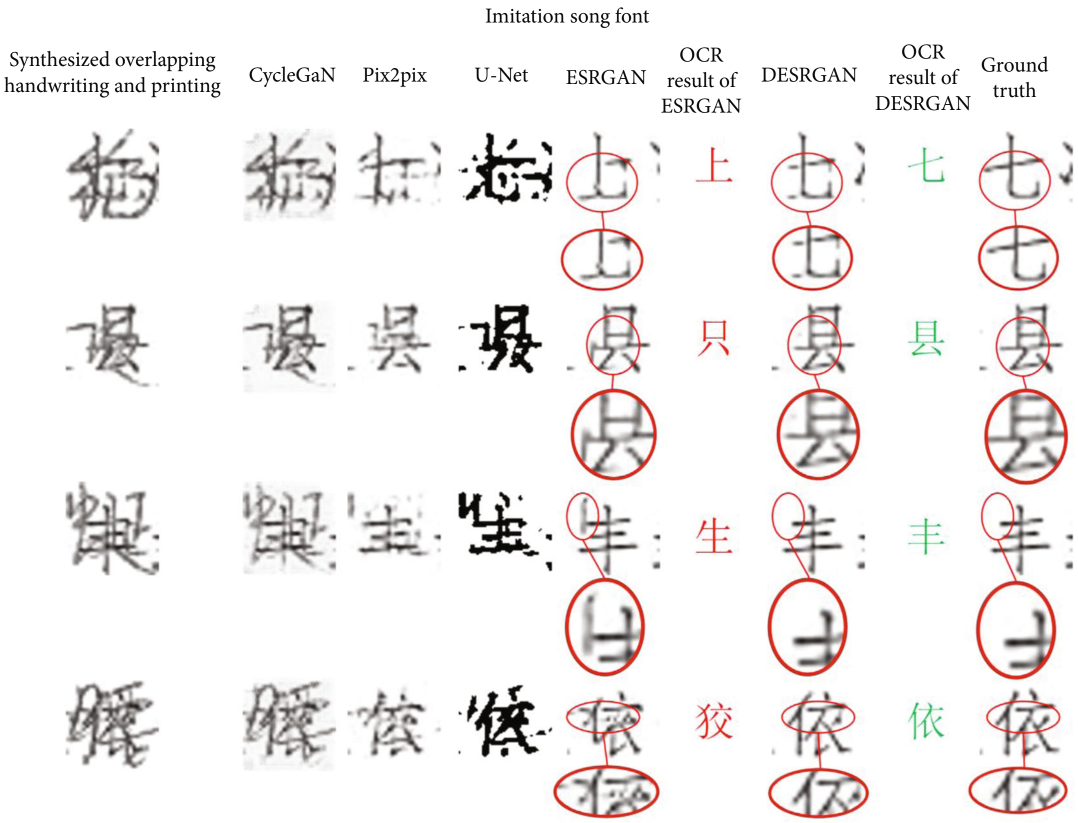


Figure 9: Results of separating printed Chinese characters in Imitation Song font. Critical stokes or fine details of Chinese characters are clipped and enlarged for further examination. Wrong recognition results by OCR tool are colored in red while the correct ones in green.

used a network structure similar to ESRGAN [19] as our backbone network, with major modifications. The dense connection structure of the ESRGAN generator network directly transfers the Chinese character strokes and skeleton information extracted from the intermediate layer to the subsequent layer. The proposed DESRGAN generator network removed the original upsampling layer of the ESRGAN and further replaced original convolution kernels with dilated convolution kernels. VGG19 was used to implement the discriminator which validates whether the image generated by the generative network is real or fake.

In ESRGAN, the loss function is a weighted sum of three components: a perceptual loss L_{percep} which measures the distance between the separated image and the ground truth image features before activation in pretrained VGG19, an adversarial loss L_G^{Ra} based on the probabilities of a relativistic discriminator, and a content loss $L_1 = \mathbb{E}_{x_i} \| G(x_i) - y \|_1$ which evaluates the 1-norm distance between separated printed or handwritten character image $G(x_i)$ and ground truth y.

The perceptual loss L_{percep} is defined as

$$L_{\text{percep}} = \left\| \phi(x_r) - \phi\left(x_f\right) \right\|_1, \tag{1}$$

where $\phi(\cdot)$ represents features before activation in pretrained VGG19, x_r stands for clean printed or handwritten character image, $x_f = G(x_i)$, and x_i stands for the mixed image of handwritten and printed characters.

The adversarial loss for generator is defined as

$$L_G^{\text{Ra}} = -\mathbb{E}_{x_r}\left[\log\left(1 - D_{\text{Ra}}\left(x_r, x_f\right)\right)\right] - \mathbb{E}_{x_f}\left[\log\left(D_{\text{Ra}}\left(x_f, x_r\right)\right)\right], \tag{2}$$

where $D_{\text{Ra}}(x_r, x_f) = \sigma(C(x_r) - \mathbb{E}[C(x_f)])$, σ is sigmoid function, $C(x)$ is the nontransformed discriminator output, and $\mathbb{E}_{x_f}[\cdot]$ represents the operation of taking average for all fake data in the mini-batch.

Perceptual loss L_{percep} plays an important role in computer vision tasks such as super resolution where the richness of the details of the recovered image is critical. It is designed to improve high-frequency details and avoid blurry and unpleasant visual effects. However, the goal we want to achieve here is to separate the printed part from the overlapped handwriting as much as possible and indirectly improve the recognition accuracy of subsequent OCR. With regard to this, we believe that the overall structure of the character and the integrity of the strokes are more important than the high-frequency details for OCR tools. Take the case in Figure 3 for example, due to the lack of a stroke in the center of the recovered image, OCR tools output the character “九” other than the correct one “丸”.

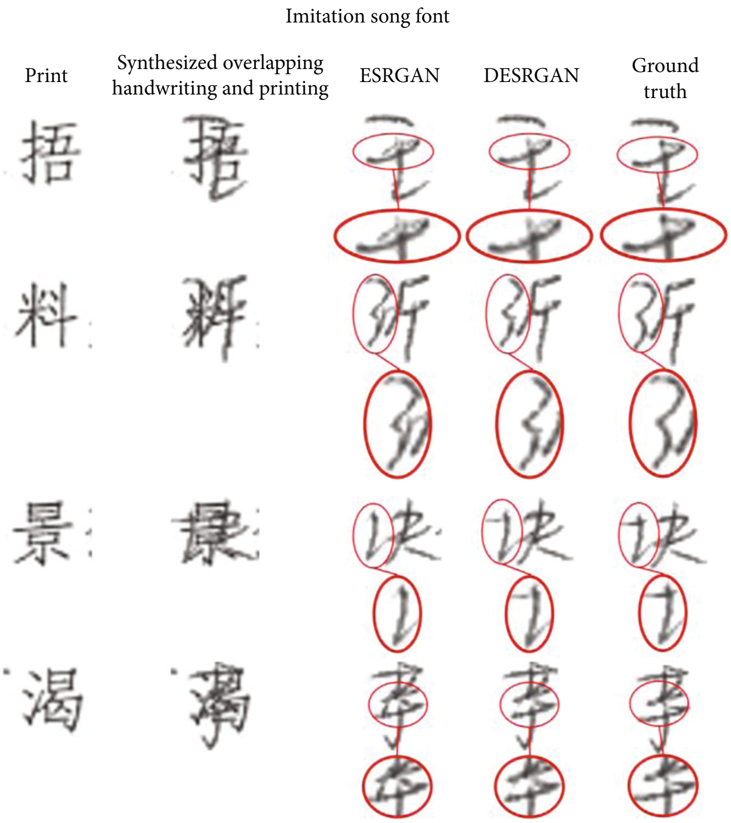


Figure 10: Results of separating handwritten Chinese characters from superimposed printed characters in Imitation Song font. Columns from left to right: printed characters, synthesized handwritten/printed overlapped data sample in test set, results of ESRGAN and DESRGAN, and ground truth. Stokes or fine details of reconstructed handwriting are clipped and enlarged for further examination.

Therefore, a novel gradient-based loss term that can measure the integrity of the font skeleton was explored. Image gradients are powerful shape features, widely used in computer vision tasks. Given an overlapping image I^{OL}, the recovered image $G(I^{\mathrm{OL}})$, and its ground truth I^{GT}, the gradients of $G(I^{\mathrm{OL}})$ and I^{GT} were calculated, denoted as $\nabla G(I^{\mathrm{OL}})$ and ∇I^{GT}, respectively. Instead of relying on whole image level losses, we build on the ideas of gridding and max-pooling. As seen in Figure 3, the whole image area is divided into a square gird (3×3 or 5×5) of cells $\{C_i\}$, and the integrity loss was defined as the largest mean square error between $\nabla G(I^{\mathrm{OL}})$ and ∇I^{GT} of each cell C_i

$$L_{\text{integrity}} = \max_{C_i} \frac{1}{W_i H_i} \sum_{x,y\in C_i} \left\| \nabla I^{\mathrm{GT}}_{x,y} - \nabla G\left(I^{\mathrm{OL}}\right)x, y \right\|^2, \qquad (3)$$

where W_i and H_i are the width and height of cell C_i. With this strategy, the integrity of every cell of the skeleton is evaluated. The integrity loss will locate the cell with severe discrepancy between the recovered strokes and the ground truth strokes.

Therefore, the total loss for the generator is

$$L_G = L_{\text{percep}} + \lambda L_G^{\text{Ra}} + \eta L_1 + \alpha L_{\text{integrity}}, \qquad (4)$$

where λ, η, and α are the coefficients to balance different loss terms.

4. Experiment Settings

4.1. Dataset. In this work, we focus on character-level separation techniques. To process a document, some existing related technologies, such as layout analysis [27–29] and connected-component analysis [30], can help locate character positions. Therefore, we assume that there are some front-end modules that can help us roughly segment printed characters from a complete document. For the experiment, a dataset containing only overlapping printed and handwritten characters was created, as described below.

The handwritten character images used for synthesis come from the CASIA HWDB (CASIA Handwritten Database) 1.1 dataset [31], which contains images of 3755 commonly used Chinese character images written by 300 different writers. Specifically, we randomly chose handwritten images from four different writers (writer IDs 1003, 1062, 1187, and 1235) for synthesis.

The printed character images used in the synthesis include images of the same 3,755 commonly used Chinese characters listed in CASIA HWDB 1.1. In addition, basic symbols (plus sign, minus sign, equal sign, and answer box)

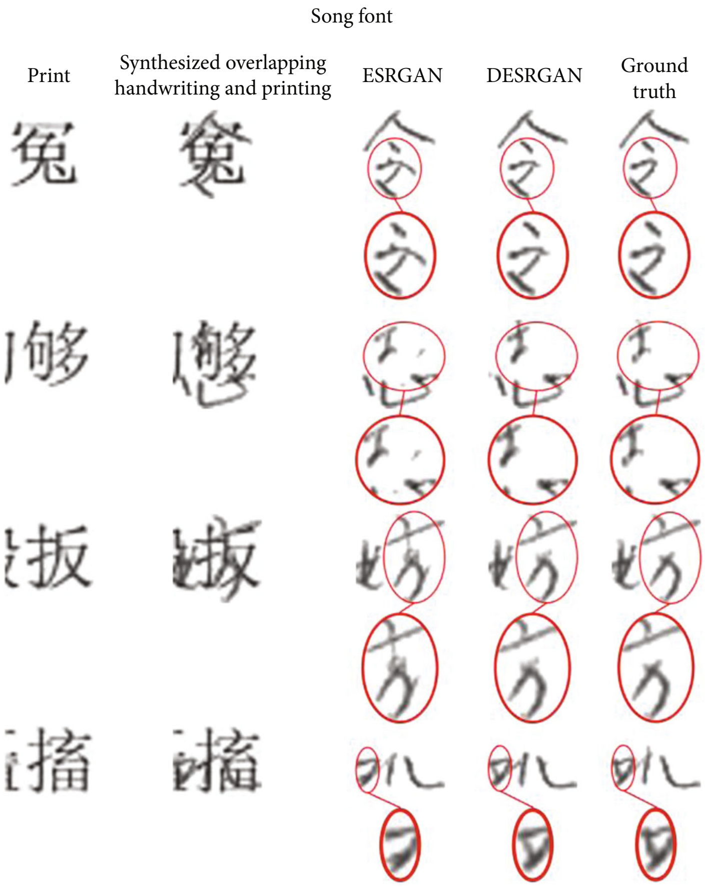


Figure 11: Results of separating handwritten characters from superimposed printed characters in Song font.

and Arabic numerals 0 to 9 were also added to the printed image dataset, which contains a total of 3,769 characters. For those 3,769 characters, they were printed on the A4 size paper in four fonts (Song, Hei, Kai, and Imitation Song). The printed paper was scanned and transferred to an image. Then, the scanned image was cropped at the character level to obtain the printed images for synthesis.

Existing researches on the pixel level separation of handwritten characters and printed characters are few. There is currently no publicly available dataset of overlapped handwritten and printed characters with pixel-level annotations. Therefore, this work uses handwritten character images and printed character images to synthesize samples of handwritten characters overlapped with printed characters. As Figure 4 shows, the method of image synthesis is to calculate the minimum value of the gray value of the pixels of the two images at the same position and use this minimum value as the gray value of the corresponding pixel of the composite picture.

The selected handwriting samples are paired with the printed samples by random matching. The final pairing results are as follows: printed Hei is randomly paired with the handwritten from CASIA HWDB writer ID 1003; printed Song is randomly paired with the handwritten from CASIA HWDB writer ID 1062; printed Kai randomly paired with the handwritten from CASIA HWDB writer ID 1187; and printed Imitation Song randomly paired with the handwritten from CASIA HWDB writer ID 1235.

On this basis, we refer to a 708 Chinese character set that contains various basic components of Chinese characters proposed by Lian et al. [10] in the study of handwritten Chinese character synthesis [10] (the collection of Chinese characters in their work contains a total of 775 characters, of which there are 67 unusual characters that are not in the CASIA HWDB dataset). Images containing these 708 Chinese characters also constitute the training dataset during the training phase. We assume that the model only needs to learn the features of these 708 Chinese characters to achieve the separation goal, rather than learning the features of all 3769 characters. Reducing the number of characters in the training set has a beneficial effect on both data collection effort and computation cost. Therefore, shown in Figure 5, 708 corresponding samples from handwriting samples and printed samples were selected for random pairing synthesis as the training set. The remaining samples are also taken as a test set by random pair synthesis. The resulting training set contains 2832 samples (708 samples for each font type), and the test set contains 12244 samples. This dataset is used as the dataset commonly used in all subsequent experiments.

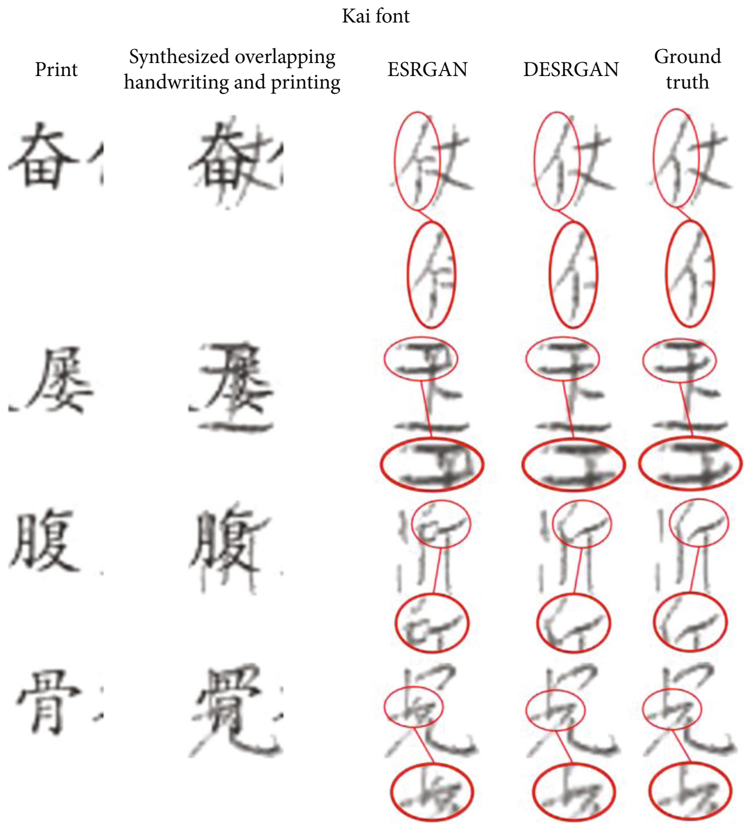


Figure 12: Results of separating handwritten characters from superimposed printed characters in Kai font.

4.2. Evaluation Metrics. Both intersection over Union (IoU) and OCR accuracy were used as our evaluation metrics. The separation of handwritten characters and printed characters can essentially be regarded as the semantic segmentation of a mixed image of handwritten and printed characters. The most commonly used quantitative evaluation metric in image semantic segmentation is IoU. Therefore, IoU was used as one of the quantitative evaluation metrics for evaluating the separation quality of handwriting and printed characters.

Because this study is a pixel-level segmentation of handwritten printed mixed pictures, IoU is calculated by the number of pixels in the corresponding category (i.e., background and printed). Before calculating IoU, the image is first binarized, and the black area after binarization is regarded as the printed area, and the white area is regarded as the background area. In the process of binarization, the Otsu algorithm is first used to calculate the average binarization threshold of all printed samples in the test set, and this single threshold value is used for binarization of all samples. In our synthesized dataset, the average binarization threshold calculated by the Otsu algorithm for the test set is 184. Finally, we divide the intersection of the separated printed part (or background) and the printed part (or background) in ground truth by their union.

One of applications targeted by this study is the automatic grading of exam papers for primary and middle school students. Therefore, the main purpose of separating handwritten characters from printed characters in this research is to improve the recognition accuracy of the printed text and handwritten text and to prevent the deterioration of recognition of the printed text due to the interference of handwritten strokes or characters on the exam paper. Therefore, the accuracy of OCR for printed characters is used as another quantitative evaluation metric for the separation of handwritten and printed characters.

The OCR tool used to calculate the accuracy of OCR is Chinese_OCR [32], which is open sourced on Github. This model recognizes Chinese characters with high speed and high accuracy. It is very suitable for evaluating the accuracy of OCR of the separated printed samples. When calculating the accuracy of OCR, because Chinese_OCR cannot detect the text of a single character image, we had to first horizontally concatenate every 25 samples into a long image for OCR. Then, the correct number of characters was counted and identified according to the character order in the long image.

4.3. Model Training. The experiment was conducted on a PC with Intel Xeon E5-2603 v3@1.600GHz CPU, NVIDIA Tesla P40 24GB GPU, and 64GB memory. The PC runs the CentOS 7 operating system, and the deep learning framework used is PyTorch 1.2.0.

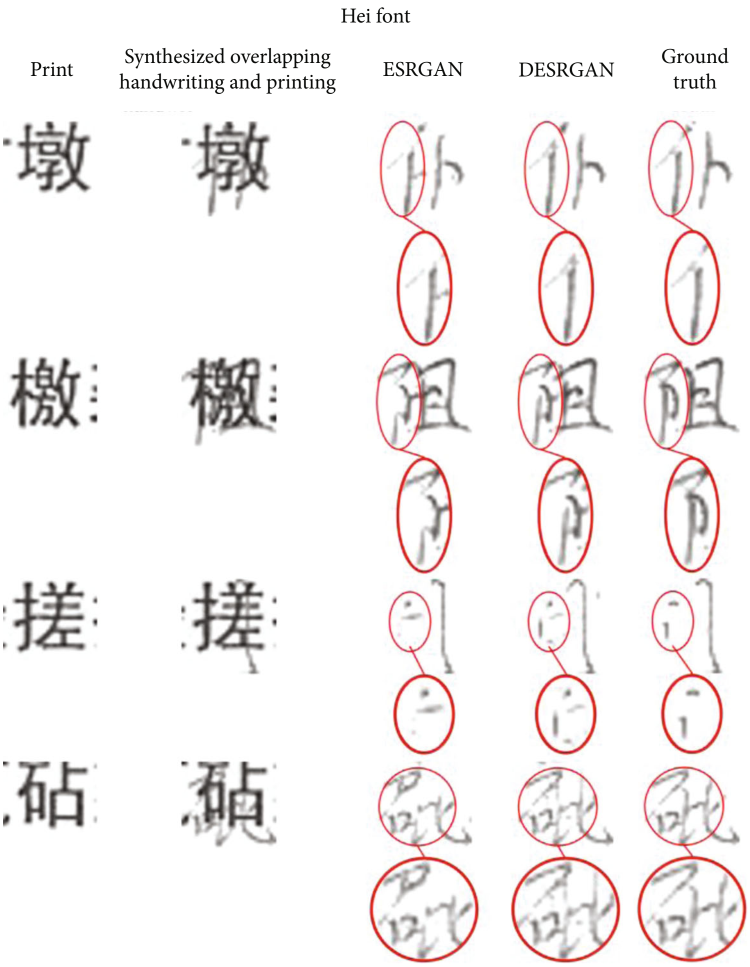


Figure 13: Results of separating handwritten characters from superimposed printed characters in Hei font.

In order to verify the effectiveness of the DESRGAN model, U-Net, Pix2pix, and CycleGAN which are commonly used in image semantic segmentation were selected as a comparison. Both Pix2pix and CycleGAN models were trained for 200 epochs with the batch size set to 1. The learning rate of the first 100 epochs remained at 0.0002, and the learning rate of the last 100 epochs decayed linearly to 0. U-Net was trained for 200 epochs with the batch size set to 100. The learning rate was initially 0.01 and then dropped to tenth of its value after every 50 epochs.

At the same time, we compared with ESRGAN to verify the effectiveness of the proposed modification to ESRGAN. Both ESRGAN and DESRGAN first used L1 loss to train their generators for 2500 epochs separately with the batch size set to 16. The initial learning rate was 0.0002 and then halved after every 200,000 iterations. Then, combined with the discriminator network, the training method of the GAN is used to train 2500 epochs with the batch size set to 16. The initial learning rate was 0.00001 and halved after the 50,000th, 100,000th, 200,000th, and 300,000th iterations. Throughout our work, the coefficient λ of adversarial loss L_G^{Ra} and the coefficient η of content loss L_1 are set to 0.005 and 0.01, respectively.

Table 1: IoU of the separation results of printed characters by different deep learning methods. The IoU of printed and the IoU of background are first calculated for each result against ground truth in the test set and then averaged. The overall IoU is the average of the first two values.

IoU	CycleGAN	Pix2pix	U-Net	ESRGAN	DESRGAN
Printed	0.631	0.755	0.697	0.905	0.911
Background	0.869	0.935	0.910	0.977	0.978
Overall	0.750	0.845	0.803	0.941	0.944

5. Results and Discussion

5.1. Visual Effects of Separation Results. We verified several deep learning methods including our proposed DESRGAN and visually compared the separation results of Chinese characters in the test set. To understand the performance of separating printed Chinese characters from noisy background, we synthesized handwritten/printed overlapped data samples and tested five methods (i.e., CycleGAN, Pix2pix, U-Net, ESRGAN, and DESRGAN). Figures 6–9 show the results of these five methods, along with the separation ground truth.

Table 2: Impact of the proposed loss function on IoU of the separated printed characters.

IoU	DESRGAN without $L_{\text{integrity}}$	DESRGAN + $L_{\text{integrity}}$ ($\alpha = 0.1$)	DESRGAN + $L_{\text{integrity}}$ ($\alpha = 1$)	DESRGAN + $L_{\text{integrity}}$ ($\alpha = 10$)
Printed	0.911	0.913	0.912	0.910
Background	0.978	0.979	0.978	0.978
Overall	0.944	0.946	0.945	0.944

To further understand the impact of a slight loss or increase of strokes in the separation result, the recognition results of OCR tool were additionally placed next to the separation results.

Four popular printed Chinese font types were tested: Hei, Kai, Song, and Imitation Song. For all tested font types, ESRGAN and DESRGAN gave the most visually pleasing results. As shown in Figures 6–9, other methods could not completely eliminate handwriting strokes in the separation result. However, not all fonts are designed equal and some of them (i.e., Song font and Imitation Song font) contain much thinner strokes. As a result, ESRGAN failed to reconstruct some seemly trivial strokes or remove artifacts which are harder to distinguish from the superimposed handwriting. The more complex the structure of Chinese characters or the greater the possibility of similar structures, the easier it is for OCR tools to predict seemly correct but substantially wrong results. Only DESRGAN gave separation results that produced most successful OCR predictions (see Figures S1-S4 in the Supplementary Material for more separation results of printed characters).

Since DESRGAN can effectively separate the printed part from the overlapping image, the next question is whether it is capable of the separating the handwritten part. In essence, this task is more difficult because the handwriting style varies from person to person. For this test, only ESRGAN and DESRGAN were compared because the other three methods produce poor results. We did not report recognition results because no suitable OCR tool for handwriting recognition was found. Figures 10–13 show the visual effect of separating handwritten parts from superimposed printed characters in four different font types. The best separation effect came from the Imitation Song font (Figure 10) and the Song font (Figure 11), while the Hei font gave the worst effect (Figure 13). We speculated that it is because the strokes of printed characters in Hei font are thicker and the colors are darker, which interfered more with handwritten characters. Nonetheless, DESRGAN produced less artifacts and reconstructed better character structure than ESRGAN (see Figures S5-S8 in the Supplementary Material for more separation results of handwriting).

5.2. Quantitative Analysis. As shown in Table 1, the IoU of the separation results of printed characters confirmed the previous visual effect comparison. Both ESRGAN and DESRGAN achieved better results than other deep learning methods, and DESRGAN has a small advantage over ESRGAN. It should be noted that through visual analysis, DESRGAN is better at restoring important details, which only account for a small part of the total pixels.

Table 3: IoU of the separation results of handwritten characters.

	ESRGAN	DESRGAN
IoU of handwriting	0.830	0.834
IoU of background	0.952	0.953
Overall IoU	0.891	0.894

In order to study the impact of the proposed loss function, we conducted experiments under three different settings (i.e., $\alpha = 0.1$, $\alpha = 1$, and $\alpha = 10$). Table 2 shows that the proposed loss function had almost no impact on IoU. This is in line with our expectations, because the main purpose of the new loss function is to improve the overall structure of the characters.

The IoU of separation of handwritten characters by ESRGAN and DESRGAN was also evaluated. Both models received more than 10,000 overlapping Chinese characters from which the handwriting parts were reconstructed individually. The IoU of separated handwriting and the IoU of separated background were first calculated for each result against ground truth in the test set. Table 3 shows that DESRGAN achieved slightly better IoU results.

Table 4 shows that the superimposed handwriting has a great negative impact on the accuracy of OCR. The worst synthesized overlapping in Imitation Song font only achieved zero accuracy. The separation results of ESRGAN and DESRGAN led to higher OCR accuracy than those of CycleGAN, Pix2pix, and U-Net, which proved the advantage of network structure in identifying characters from a noisy background. Furthermore, DESRGAN achieved the highest OCR accuracy in three fonts (i.e., Kai, Song, and Imitation Song) thanks to better preservation of the strokes and basic skeleton of Chinese characters. Compared to ESRGAN, the proposed method improved the OCR accuracy by more than 1% in Song font and Imitation Song font which are more difficult to handle due to their thin strokes and light colors after scanning. Except Imitation Song font, the recognition accuracy rate of the OCR tool for the separation results of DESRGAN has almost reached the level of recognition of ground truth.

The impact of the proposed loss function on OCR accuracy was also measured. As shown in Table 5, the proposed loss component improved the OCR accuracy, especially in the case of Imitation Song font. This result coincides with previous visual analysis, where the OCR results of Imitation Song font are susceptible to trivial loss in characters most.

Table 4: Recognition result of separated printed characters and ground truth by OCR tool.

	Hei font	Kai font	Song font	Imitation Song font	Average
Synthesized overlapping	0.288	0.001	0.020	0.000	0.077
CycleGAN	0.615	0.063	0.129	0.028	0.209
Pix2pix	0.909	0.831	0.507	0.345	0.648
U-Net	0.622	0.587	0.296	0.217	0.431
ESRGAN	0.966	0.962	0.925	0.889	0.936
DESRGAN	0.964	0.971	0944	0.903	0.945
Ground truth	0.968	0.981	0.976	0.981	0.977

Table 5: Impact of the proposed loss function on OCR accuracy of separated printed characters.

	Hei font	Kai font	Song font	Imitation Song font	Average
DESRGAN without $L_{\text{integrity}}$	0.964	0.971	0.944	0.903	0.945
DESRGAN + $L_{\text{integrity}}$ ($\alpha = 0.1$)	0.965	0.972	0.947	0.912	0.949
DESRGAN + $L_{\text{integrity}}$ ($\alpha = 1$)	0.964	0.972	0.936	0.910	0.946
DESRGAN + $L_{\text{integrity}}$ ($\alpha = 10$)	0.962	0.971	0.948	0.912	0.948

6. Conclusions

In summary, a method to separate Chinese characters from noisy background where other characters are likely to overlap was proposed. Our method reconstructed important strokes and retained the overall structure in the complex Chinese characters. The proposed method also allowed the OCR tool to achieve better recognition accuracy. Those findings may have great benefits to scenarios such as test paper autoscoring and advanced document analysis. Our future works include studying how color of handwriting impacts separation process and applying to other writing system.

Data Availability

Data of printed Chinese characters of four common fonts is available upon request. You may contact the corresponding author to request data. We do not own the dataset of handwritten Chinese characters (CASIA HWDB), and you may send data request to the owner.

Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

Acknowledgments

This research was funded in part by Natural Science Foundation of Fujian Province of China (grant numbers 2019J01712, 2018H0025, and 2019J05099), in part by Xiamen Science and Technology Bureau (grant numbers 3502Z20183035, 3502Z20183037, and 3502Z20183038), in part by Li ShangDa Fund (grant number ZC2015008), and in part by Guandong Provincial Key Laboratory of Digital Signal and Image Processing Technology Open Project (grant number 2017GDDSIPL-02). We would like to thank Kaijin Cui for data collection work. We thank Guorong Cai for their valuable comments.

Supplementary Materials

We provide more separation results in the supplementary material for reference. In the results, we show the visual effect of separating the handwritten part and the printed part, respectively, of four different common Chinese fonts. *(Supplementary Materials)*

References

[1] R. Gomez, B. Shi, L. Gomez et al., "ICDAR2017 robust reading challenge on COCO-Text," in *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, pp. 1435–1443, Kyoto, Japan, 2017.

[2] N. Nayef, F. Yin, I. Bizid et al., "ICDAR2017 robust reading challenge on multi-lingual scene text detection and script identification - RRC-MLT," in *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, pp. 1454–1459, Kyoto, Japan, 2017.

[3] R. Zhang, Y. Zhou, Q. Jiang et al., "Icdar 2019 robust reading challenge on reading Chinese text on signboard," in *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pp. 1577–1581, Sydney, NSW, Australia, 2019.

[4] J. Jo, J. W. Soh, and N. I. Cho, "Handwritten text segmentation in scribbled document via unsupervised domain adaptation," in *2019 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pp. 784–790, Lanzhou, China, 2019.

[5] X. H. Li, F. Yin, and C. L. Liu, "Printed/handwritten texts and graphics separation in complex documents using conditional random fields," in *2018 13th IAPR International Workshop on Document Analysis Systems (DAS)*, pp. 45–150, Vienna, Austria, 2018.

[6] A. Kölsch, A. Mishra, S. Varshneya, M. Z. Afzal, and M. Liwicki, "Recognizing challenging handwritten annotations with fully convolutional networks," in *2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, pp. 25–31, Niagara Falls, NY, USA, 2018.

[7] M. Alberti, L. Vögtlin, V. Pondenkandath, M. Seuret, R. Ingold, and M. Liwicki, "Labeling, cutting, grouping: an efficient text line segmentation method for medieval manuscripts," in *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pp. 1200–1206, Sydney, NSW, Australia, 2019.

[8] R. Smith, "An overview of the Tesseract OCR engine," in *Ninth international conference on document analysis and recognition (ICDAR 2007)*, vol. 2, pp. 629–633, Curitiba, Brazil, 2007.

[9] X. Y. Zhang, F. Yin, Y. M. Zhang, C. L. Liu, and Y. Bengio, "Drawing and recognizing Chinese characters with recurrent neural network," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 4, pp. 849–862, 2018.

[10] Z. Lian, B. Zhao, X. Chen, and J. Xiao, "EasyFont: a style learning-based system to easily build your large-scale handwriting fonts," *ACM Transactions on Graphics (TOG)*, vol. 38, no. 1, pp. 1–18, 2018.

[11] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: convolutional networks for biomedical image segmentation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 234–241, Cham, 2015.

[12] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3431–3440, Boston, Massachusetts, USA, 2015.

[13] L. C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "DeepLab: semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 4, pp. 834–848, 2018.

[14] C. Chu, A. Zhmoginov, and M. Sandler, "CycleGAN, a master of steganography," 2017, https://arxiv.org/abs/1712.02950.

[15] I. Goodfellow, J. Pouget-Abadie, M. Mirza et al., "Generative adversarial nets," in *Advances in neural information processing systems*, no. article 26722680, 2014ACM, New York, 2014.

[16] J. Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 2242–2251, Venice, Italy, 2017.

[17] P. Isola, J. Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5967–5976, Honolulu, Hawaii, USA, 2017.

[18] C. Ledig, L. Theis, F. Huszár et al., "Photo-realistic single image super-resolution using a generative adversarial network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4681–4690, Honolulu, Hawaii, USA, 2017.

[19] X. Wang, K. Yu, S. Wu et al., "Esrgan: enhanced super-resolution generative adversarial networks," in *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, Munich, Germany, 2018.

[20] S. Brunessaux, P. Giroux, B. Grilhères et al., "The maurdor project: improving automatic processing of digital documents," in *2014 11th IAPR International Workshop on Document Analysis Systems*, pp. 349–354, Tours, France, 2014.

[21] W. R. Huang, Y. Qi, Q. Li, and J. Degange, "DeepErase: weakly supervised ink artifact removal in document text images," 2019, https://arxiv.org/abs/1910.07070.

[22] J. K. Guo and M. Y. Ma, "Separating handwritten material from machine printed text using hidden Markov models," in *Proceedings of sixth international conference on document analysis and recognition*, pp. 439–443, Seattle, WA, USA, 2001.

[23] K. Zagoris, I. Pratikakis, A. Antonacopoulos, B. Gatos, and N. Papamarkos, "Distinction between handwritten and machine-printed text based on the bag of visual words model," *Pattern Recognition*, vol. 47, no. 3, pp. 1051–1062, 2014.

[24] A. Lat and C. V. Jawahar, "Enhancing OCR accuracy with super resolution," in *2018 24th International Conference on Pattern Recognition (ICPR)*, pp. 3162–3167, Beijing, China, 2018.

[25] A. Graves, "Generating sequences with recurrent neural networks," 2013, https://arxiv.org/abs/1308.0850.

[26] B. Chang, Q. Zhang, S. Pan, and L. Meng, "Generating handwritten Chinese characters using CycleGAN," in *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 199–207, Lake Tahoe, NV, USA, 2018.

[27] Y. Li, Y. Zou, and J. Ma, "Deeplayout: a semantic segmentation approach to page layout analysis," in *International Conference on Intelligent Computing*, pp. 266–277, Cham, 2018.

[28] X. Yang, E. Yumer, P. Asente, M. Kraley, D. Kifer, and C. Lee Giles, "Learning to extract semantic structure from documents using multimodal fully convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5315–5324, Honolulu, Hawaii, USA, 2017.

[29] Y. Xu, F. Yin, Z. Zhang, and C. L. Liu, "Multi-task layout analysis for historical handwritten documents using fully convolutional networks," in *International Joint Conference on Artificial Intelligence*, pp. 1057–1063, Stockholm, Sweden, 2018.

[30] F. Chang, C. J. Chen, and C. J. Lu, "A linear-time component-labeling algorithm using contour tracing technique," *Computer Vision and Image Understanding*, vol. 93, no. 2, pp. 206–220, 2004.

[31] C. L. Liu, F. Yin, D. H. Wang, and Q. F. Wang, "CASIA online and offline Chinese handwriting databases," in *2011 International Conference on Document Analysis and Recognition*, pp. 37–41, Beijing, China, 2011.

[32] YCG09, *Chinese_OCR*October 2019, https://github.com/YCG09/chinese_ocr.