Wireless Communications and Mobile Computing

Developments in Mobile Multimedia Technologies 2022

Lead Guest Editor: Yuanlong Cao
Guest Editors: Cunhua Pan, Wei Quan, and Xun Shao

Developments in Mobile Multimedia Technologies 2022

Wireless Communications and Mobile Computing

Developments in Mobile Multimedia Technologies 2022

Lead Guest Editor: Yuanlong Cao
Guest Editors: Cunhua Pan, Wei Quan, and Xun Shao

Chief Editor

Associate Editors

Academic Editors

Contents

Hindawi
Wireless Communications and Mobile Computing
Volume 2023, Article ID 8466088, 12 pages
https://doi.org/10.1155/2023/8466088

Research Article

Image Compression for Wireless Sensor Network: A Model Segmentation-Based Compressive Autoencoder

Xuecai Bao,[1,2] **Chen Ye**,[1,2] **Longzhe Han**,[1,2] **and Xiaohua Xu**[3]

[1]*Jiangxi Province Key Laboratory of Water Information Cooperative Sensing and Intelligent Processing, Nanchang Institute of Technology, 330099 Nanchang, Jiangxi, China*
[2]*School of Information Engineering, Nanchang Institute of Technology, 330099 Nanchang, Jiangxi, China*
[3]*Jiangxi Academy of Water Science and Engineering, 330029 Nanchang, Jiangxi, China*

Correspondence should be addressed to Xuecai Bao; lx97821@126.com

Received 12 August 2022; Revised 27 September 2022; Accepted 16 February 2023; Published 25 October 2023

Academic Editor: Yuanlong Cao

Aiming at the problems of image quality, compression performance, and transmission efficiency of image compression in wireless sensor networks (WSN), a model segmentation-based compressive autoencoder (MS-CAE) is proposed. In the proposed algorithm, we first divide each image in the dataset into pixel blocks and design a novel deep image compression network with a compressive autoencoder to form a compressed feature map by encoding pixel blocks. Then, the reconstructed image is obtained by using the quantized coefficients of the quantizer and splicing the decoded feature maps in order. Finally, the deep network model is segmented into two parts: the encoding network and the decoding network. The weight parameters of the encoding network are deployed to the edge device for the compressed image in the sensor network. For high-quality reconstructed images, the weight parameters of the decoding network are deployed to the cloud system. Experimental results demonstrate that the proposed MS-CAE obtains a high signal-to-noise ratio (PSNR) for the details of the image, and the compression ratio at the same bit per pixel (bpp) is significantly higher than that of the compared image compression algorithms. It also indicates that the MS-CAE not only greatly relieves the pressure of the hardware system in sensor network but also effectively improves image transmission efficiency and solves the deployment problem of image monitoring in remote and energy-poor areas.

1. Introduction

The wireless sensor network (WSN) is widely deployed in many applications, such as ecological environment monitoring, water quality monitoring, and mine safety monitoring [1–4]. Image monitoring in WSN is an important topic in the monitoring field. It has a visual effect and can provide image information to the management platform. However, the massive amounts of image information cause network congestion. Although some novel technologies of congestion control and packet reordering algorithms are proposed to solve this problem [5–7], the image compression technology in the image sensor device has attracted an increasing attention and is considered as an effective solution in terms of improving energy and transmission efficiency. Until now, many image compression algorithms for WSN have been proposed [8]. However, owing to the functional limitations of hardware equipment for WSN and the high energy consumption of image transmission, it also poses significant challenges to WSN deployment in remote areas with limited energy.

For traditional image compression techniques in WSN, the research on image compression can be categorized as lossless and lossy image compression. JPEG [9] and JPEG 2000 [10] are typical representations of lossy image compression and have been widely applied to WSN. Aiming at transmission efficiency and memory saving, lossy image compression draws more attention in WSN than lossless image compression. In particular, the emergence of image compression techniques based on Deep Learning Models (DLMs) provides a completely new direction [11].

In the field of deep learning image compression, a great number of efforts have been devoted to improving the resolution of reconstructed compressed images. Using the Convolutional Neural Network (CNN) structure, the methods of training Compact CNN (ComCNN) and Reconstructed CNN (RecCNN) are proposed simultaneously in [12]. ComCNN mainly optimizes the compression effect, and ReCNN is used to reconstruct high-quality images. Kuang et al. propose a new model for a single-image Super-Resolution (SR) task by utilizing the design of densely connected convolutional networks (DenseNet) [13], which has a lightweight structure and is extensively evaluated on datasets. They attempt to optimize the deep network and adjust parameter settings to achieve trade-offs between image resolution and running time. The advantage of deep CNNs lies in its powerful capability to handle large-scale image datasets. These works, on the other hand, are complex, making them difficult to deploy in WSN edge devices.

Currently, autoencoders based on CNN have become a significant research interest, which are more simple than deep CNN in network architecture. In an earlier period, most learning autoencoders were used for dimensionality reduction for high-efficiency image compression. On the other hand, the autoencoder, with its relatively simple network architecture, is also faster than CNN in the inference process. Huang et al. propose a multiscale autoencoder (MSAE) to improve the compression effect and adopt the generative adversarial network (GAN) with multiscale discriminators to perform the end-to-end trainable rate-distortion optimization. This framework achieves excellent reconstruction effects at a low bit rate [14]. Cheng et al. use Principal Component Analysis (PCA) to generate an energy-efficient representation for the CAE architecture to achieve high coding efficiency, and the algorithm mainly preserves the principal components in the model training process and greatly improves the compression ratio [15]. Furthermore, when compared to the traditional deep CNN architecture, CAE-based image compression is a complete deep learning architecture that reduces its own network layers [16]. Based on an autoencoder, the authors in [17] append quantization and entropy rate estimation to the CNN structure. Furthermore, in [18], a three-dimensional convolutional autoencoder (3D-CAE) is proposed, which has greatly improved the reconstruction precision. All these algorithms mentioned above improve the network architecture of the compressive autoencoder, which performs well in reconstructed image detail extraction. In addition, the end-to-end architecture also offers the possibility of deployment for WSN. However, some of these algorithms will occupy a great deal of memory at runtime, which impacts the efficiency of image monitoring for WSN.

Moreover, most of the above-mentioned works focus on the optimization of rate distortion, visual effect, and image compression ratio, but the limited memory capacity of the hardware system in the WSN is not considered. Aiming to solve the above-mentioned problems, we propose a novel MS-CAE algorithm to satisfy the demands of WSN image monitoring in remote areas. The main contributions of the proposed MS-CAE algorithm areas are as follows:

(1) To address the issue of large networks not being deployed in sensor nodes due to functional constraints, we proposed a model segmentation-based compressive autoencoder

(2) We proposed an asymmetric architecture for the encoding and decoding networks in MS-CAE. We design the simplified encoding network and the more complex decoding network properly to improve the resolution of the reconstructed compressed image

The rest of this paper is organized as follows: Section 2 describes the related work of image compression. Section 3 presents the principles of the architecture of a compressive autoencoder (CAE). In Section 4, we present a novel MS-CAE image compression algorithm for image monitoring in WSN. Section 5 evaluates the performance of the proposed MS-CAE algorithm, followed by concluding remarks in Section 6.

2. Related Work

2.1. Image Compression Based on Deep Learning. Recent works on the CNN network have made contributions to image compression, especially in DLMs. To achieve high-quality image compression at low bit rates, Jiang et al. propose two CNNs as the pre- and postprocessing steps [12]. Toderici et al. utilize a long short-term memory (LSTM) recurrent network to compress small patch images and also adopt quantization to realize the decrease in the encoding coefficient scale [19]. Li et al. are motivated by the character of the local information content in a single image, and they propose learning convolution networks for content-weight image compression to solve the problem of encoder rate distortion [20]. The DSSLIC framework is used to obtain the semantic segmentation map of the input image and encode it as the base layer of the bitstream [21]. Sushma and Fatimah improve the reconstructed image detail information by predicting chroma at the decoder, which serves as side information for decoding chroma components [22]. These algorithms optimize the quality of reconstructed images in various aspects. For instance, these authors make great progress in the aspects of high compression ratio, compression efficiency, high-resolution image, and detail image reconstruction, whereas the operations mentioned before usually consume a large amount of storage space in computer equipment.

2.2. Image Compression Based on CAE. There exist numerous works on variants of compressive autoencoders (CAE). In different ways, these techniques reduce the distortion of the reconstructed image for lossy image compression. In [23], Shi et al. introduce an efficient subpixel convolution layer learned from an array of upscaling filters to upscale the final low-resolution feature maps into the high-resolution output image. Inspired by the work of Shi et al. [23], Theis and Shi [16] utilize the CAE structure by optimizing quantization and entropy rate estimation to acquire excellent training model results. Following the above architectures, the authors

in [17] append a nonlinear analysis transformation, a uniform quantizer, and a nonlinear synthesis transformation to a convolutional network. Cheng et al. train the improved CAE architecture to generate a more compact representation of feature maps, and they optimize the rate-distortion loss function of CAE to improve image-coding efficiency [15]. An energy compaction-based image compression using a convolutional autoencoder is proposed. This work optimizes the CAE architecture by decomposing it into several down- and upsample operations and proposes a normalized coding gain metric in neural networks [24]. Based on the previous high-precision CAE, Chong et al. [18] exploit a 3D-CAE architecture that precisely achieves end-to-end joint spectral-spatial compression and reconstruction. These works in the literature [15, 18, 24] primarily employ a compact compression network and various upsample operations to trade-off the optimization of the compression ratio and rate distortion.

2.3. Image Compression Work in the Field of WSN. Efficient DLM models will be applicable to the interconnection between hardware systems and cloud devices. From the requirements of image monitoring, our work is divided into two aspects: edge devices and cloud-based devices. An edge device is used to obtain image information [25] and cloud-based device analysis image-coding coefficients [26]. Ding et al. deploy DLMs to edge devices and cloud-based devices, which advance the running speed of the corresponding device [27]. However, high-performance DLMs usually require numerous storage and computing resources, which make the deployment work difficult on an edge device. To solve this problem, many researchers attempt to improve the efficiency of DLMs by pruning the convolution layers or convolution kernels [28, 29]. Some works combine gradient-based optimization [30, 31] and residual learning [32] to implement steps to speed up inference in image compression algorithms. These works have made great progress toward obtaining excellent effects. Because a cloud-based device is deployed near monitor operators, it is technically reasonable for a decoder to obtain high-resolution images.

Through comparison and analysis, we found that the CAE architecture is suitable for image compression in WSN and presents excellent performance. Furthermore, CAE is simpler than CNN in network architecture. Therefore, we design a novel network architecture based on CAE and propose an image compression algorithm based on a model segmentation-based compressive autoencoder (MS-CAE), which not only segments the model to alleviate the pressure of the hardware system and promote the transmission efficiency of the sensor network but also improves the image quality and monitoring energy efficiency, so as to achieve the purpose of improving the energy efficiency for WSN image monitoring.

3. Architecture of Compressive Autoencoder

The network architecture of a compressive autoencoder consists of three modules: an encoder E, a decoder D, and a quantizer Q:

$$E : \mathbb{R}^N \longrightarrow \mathbb{R}^M, \tag{1}$$

$$D : \mathbb{R}^M \longrightarrow \widehat{\mathbb{R}}^N, \tag{2}$$

$$Q : e \longrightarrow Q(e). \tag{3}$$

The encoder E maps the original image $x \in \mathbb{R}$ to a latent representation $e \in E(x)$. The quantizer Q maps each element of e to $Q(e)$, which generates the quantized coefficients $\widehat{e} = Q(e)$. Then, the decoder D attempts to reconstruct the original image $\widehat{x} = D(\widehat{e})$ from the quantized coefficients $\widehat{e}$.

Figure 1 clearly illustrates the flow diagram of the CAE network. The original image is gradually compressed by the convolution layers to generate compressed data in the encoder. Then, the compressed data is quantized through the quantizer. Subsequently, the decoder reconstructs the image through the decompressed data.

To assist understanding, we assume that the original image dataset was encoded using linear mapping and a nonlinear activation function. As a result, the process of an encoder producing compressed data can be defined as

$$\mathbb{R}^M = g\left(W_{i+2}\mathbb{R}^N + b_{i+2}\right), \tag{4}$$

where $\mathbb{R}^N$ and $\mathbb{R}^M$ represent the original image and compressed data of the original image, respectively. The weight and the bias of the Conv3 layer are W_{i+2} and b_{i+2}, respectively. Moreover, the corresponding node activation function is defined as $g(\cdot)$.

After the encoding process, the quantizer transforms compressed data into decompressed data. The decoder obtains the decompressed data and calculates the reconstructed image sample. Obviously, the decoding process is the inverse of the encoding process, which is defined as

$$\widehat{\mathbb{R}}^N = g\left(W_{j+2}^T\mathbb{R}^M + b_{j+2}\right), \tag{5}$$

where $\widehat{\mathbb{R}}^N$ is the reconstructed image sample. The weight and the bias of the DeConv3 layer are W_{j+2} and b_{j+2}, respectively.

Next, we introduce the quantizer in Figure 1. The quantization is one of the approaches to decrease the complexity of encoding coefficients. The encoding network exploits the rounding function in the early period of the deep neural network. The rounding function is used to obtain the nearest integer of the coefficient. It is denoted as

$$f(x) = \text{round}(x, d), \tag{6}$$

where x and d are the coefficient and accuracy retained after the decimal point, respectively. Thus, to quantize the coefficients in more detail, Agustsson et al. in [33] adopt the uniform scalar quantizer, which is similar with the rounding function, as follows:

$$f(x_i) = \text{round}(x_i), \tag{7}$$

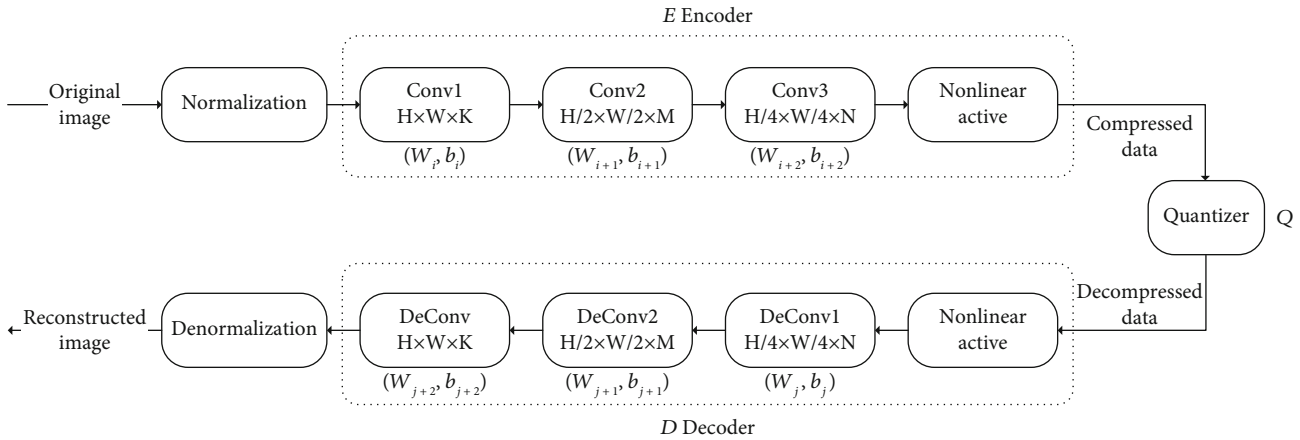


Figure 1: The network architecture of CAE.

where x and i are the coefficient and the number of equal points, respectively. Accordingly, $f(x_i)$ represents quantization through equipartition to the nearest interval.

Moreover, Toderici et al. in [19] propose a stochastic rounding function of binarization, which is written as

$$\{y\} \approx \lfloor y \rfloor + \varepsilon, \quad \varepsilon \in \{0, 1\}, \tag{8}$$

$$P(\varepsilon = 1) = y - \lfloor y \rfloor. \tag{9}$$

The stochastic rounding function is different from above-mentioned two rounding functions. The operation mainly uses the round-down method, namely, the integer of not more than x. Furthermore, the stochastic rounding function obtains round results by expectation x and random probability ε.

In the process of quantization, the rounding function exists more or less as a deviation. Therefore, rounding and the uniform scalar quantizer have more deviations. Thus, CAE uses the loss function to evaluate the train loss. From the above description, we know that the input original image sample is $X : x \in \mathbb{R}^N$, and the output reconstructed image is $\widehat{X} : \hat{x} \in \widehat{\mathbb{R}}^N$. CAE evaluates the loss rate between $\mathbb{R}^N$ and $\widehat{\mathbb{R}}^N$ by the cross-entropy loss function and mean square error (MSE) loss function. These two loss functions are defined as

$$J\left(X, \widehat{X}\right) = -\sum_{i=1}^{n} [x_i \log(\hat{x}_i) + (1 - x_i) \log(1 - \hat{x}_i)]. \tag{10}$$

$$J\left(X, \widehat{X}\right) = \frac{1}{2} \sum_{i=1}^{n} \|\hat{x}_i - x_i\|_2^2. \tag{11}$$

Following the above analysis, the loss function is minimized to acquire an excellent trained result, which is written as

$$\arg \min_{W,b} J(W, b). \tag{12}$$

4. MS-CAE Architecture and Implementation Method

In this section, we propose an image compression network architecture based on a model segmentation-based compressive autoencoder (MS-CAE) for WSN. We first present the proposed MS-CAE framework. Then, the corresponding implementation process is described. Finally, we provide the achievement of model segmentation and weight deployment for WSN.

4.1. MS-CAE Framework for WSN. The existing image compression algorithms based on CAE mainly focus on compression performance. However, few algorithms based on CAE consider the limited computing resources and the practical deployment of WSN.

Therefore, we present a novel MS-CAE framework to solve two problems:

(1) The image sensor node in the WSN cannot carry a complete trained image dataset for the deep neural network

(2) A cloud-computing platform makes it difficult to parse and reconstruct high-quality images from a simple network with insufficiently encoded data

We illustrate the proposed MS-CAE framework for WSN in Figure 2. Firstly, we divide the image dataset into several small pixel blocks by preprocessing the image. Then, the encoding network implements image compression through image feature extraction, quantification, and data compression. Subsequently, the decoding coefficients are obtained by the quantizer in the decoding network, and then, they are used to reconstruct images by the data filtering of the residual block network. In the implementation process, the obtained weight parameters by training the MS-CAE network are divided into two parts, namely, the encoding and decoding networks. Accordingly, the weight parameter information in the encoding and decoding networks is deployed to edge devices and cloud devices, respectively.

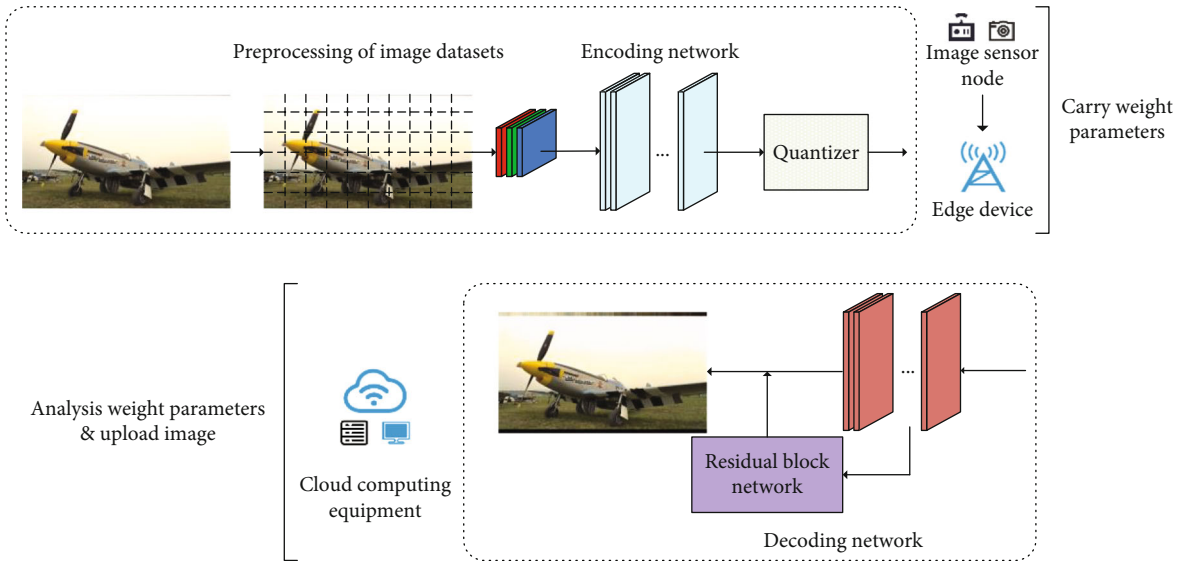


Figure 2: MS-CAE framework of image compression for WSN.

4.2. MS-CAE Network Architecture and Implementation Process

4.2.1. MS-CAE Network Architecture. The encoding and decoding networks in traditional CAE architecture are symmetrical. The symmetrical CAE architecture, on the other hand, necessitates a relatively high level of computation complexity and storage space in edge devices. It is unsuitable for an edge device with limited resources. To satisfy the demand for the edge device and cloud device, we propose a novel asymmetrical MS-CAE architecture, which is shown in Figure 3. In the proposed MS-CAE architecture, we simplify the encoding network. Meanwhile, we increase the complexity of the decoding network to improve the resolution of the reconstructed compressed image. The detailed description is as follows.

In Figure 3, after the above preprocessing data based on pixel block segmentation, each picture is decomposed into 60 three-channel (RGB) pixel blocks. The encoding and decoding networks generate three kinds of feature maps by the convolution operation. These feature maps are 128×128, 64×64, and 32×32.

In the MS-CAE network, there are five convolution kernel units. As shown in Table 1, "Conv$K$/S P" stands for a convolution layer with kernel size $K \times K$, a stride of S and a reflection padding size of P. For instance, "Conv5/2 p1.5" is a convolution unit with 5×5 convolution kernel size, 2-stride size, and 1.5 padding size.

Moreover, the reflection-padding mode is different from zero-padding. The input matrix of the reflection-padding mode is $(N, C, H_{\text{in}}, W_{\text{in}})$, and the output matrix of the reflection-padding mode is $(N, C, H_{\text{in}}, W_{\text{out}})$, where N is the number, C is the channel number, and H and W are the matrix height and width, respectively. The corresponding padding mode is written as

$$H_{\text{out}} = H_{\text{in}} + \text{padding\_top} + \text{padding\_bottom}, \tag{13}$$

$$W_{\text{out}} = W_{\text{in}} + \text{padding\_left} + \text{padding\_right}. \tag{14}$$

Furthermore, Figure 4 illustrates the zero-padding mode and the reflection-padding mode. Actually, the filled coefficients in the reflection-padding mode follow the sequence of left, right, top, and bottom. Since most deep networks adopt the zero-padding mode, the boundary pixels cannot accurately extract the coefficients through convolution operations, which causes the boundary-blurring effect. Thus, in our proposed MS-CAE, we use reflection padding to compensate for pixel gaps caused by boundary-blurring effects. Moreover, by utilizing the reflection padding in the training process, the boundary of the reconstructed image pixel blocks does not cause pixel cracks and improves the overall image quality.

4.2.2. Implementation Process

(1) Preprocessing Data: Pixel Block Segmentation. The purpose of pixel block segmentation is to divide the training images with pixel 720p ($1280 \times 720 \times 3$) into pixel 128p ($128 \times 128 \times 3$). The specific operation is as follows: We first fill the width of the image ($1280 \times 720 \times 3$ to $1280 \times 768 \times 3$). Then, the images are divided into small pixel blocks ($128 \times 128 \times 3$). Subsequently, the batches of patches are packed into the CAE-training network.

(2) Encoder Network. In the proposed MS-CAE in Figure 3, the encoder network consists of 9 convolutional layers that contain the labeled different convolution kernel units and the subsequent nonlinear operation of the parameterized rectified linear units. We adopt PReLU as an active function, which is defined as

$$\text{PReLU}(x_i) = \begin{cases} x_i, & \text{if } x_i > 0, \\ a_i x_i, & \text{if } x_i \le 0, \end{cases} \tag{15}$$

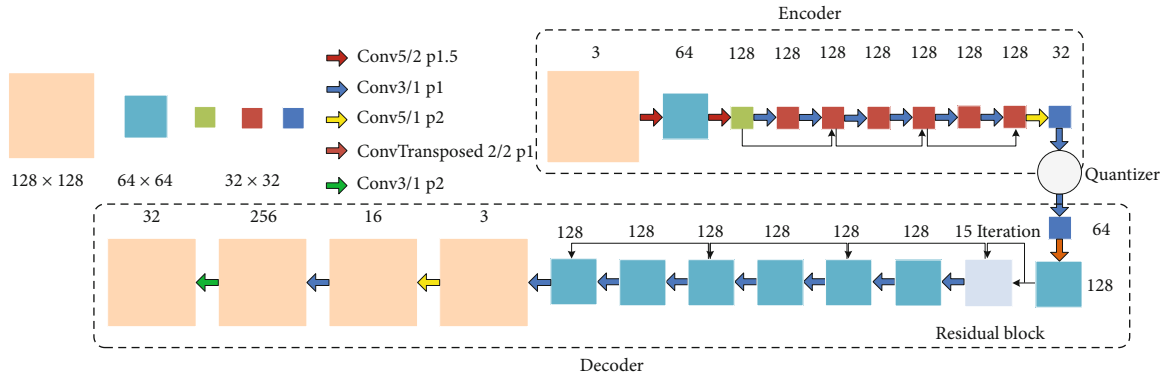


Figure 3: MS-CAE network architecture for enhancing decoder feature extraction.

Table 1: Description of different convolution kernel units.

Name	Kernel size ($K \times K$)	Stride (S)	Padding size (P)
Conv5/2 p1.5	5×5	2	1.5
Conv3/1 p1	3×3	1	1
Conv5/1 p2	5×5	1	2
Conv3/1 p2	3×3	1	2
ConvTransposed 2/2 p1	2×2	2	1

where x_i is the input of the nonlinear activation function in the matching channel i and a_i is the gradient of the negative axis of the activation function.

PReLU's nonlinear operation is conducive to the extraction and retention of negative coefficients. Through the feature linear superposition of 128 channels 32×32 with two similar Conv3/1 p1 convolution layers in three layers, the feature matrix coefficients with a lower frequency are retained as much as possible for image feature extraction.

(3) Decoder Network. The decoder of MS-CAE in Figure 3 reconstructs the 32×32 compressed feature maps obtained by the encoder. The function of the convolution layer between the encoder and the decoder network is to transform 32×32 feature blocks into 64×64 feature blocks before the process of the residual block network. As shown in Figure 3, following 15 iterations of the residual block network, 6 convolution layers are applied to increase the sample. The residual block network in the decoder relieves the gradient-vanishing problem, which efficiently avoids degradation in the next network layer.

The detailed description of the residual block network is shown in Figure 5. It consists of three convolution layers. Both the first and third convolution layers employ an 11-convolution kernel with a stride length of 1. The second layer uses a 3×3-convolution kernel with a stride length of 1. Three convolution layers are normalized and nonlinearly activated by the PReLU function. Following the filtering of the feature coefficients by the residual block network, the 64×64 feature maps of 128 channels with six Conv3/1 p1 convolution layers are used to effectively retain the nonredundant and high correlation coefficients as the foundation for reconstructing the image. Finally, the decoder obtains a reconstructed image by using 4 convolution layers.

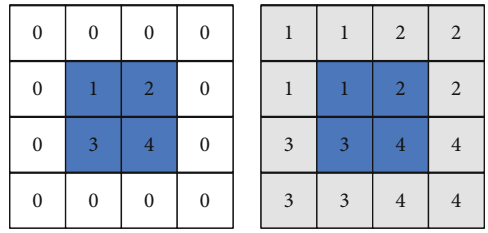
4.3. Model Segmentation and Weight Deployment for WSN. As shown in Figures 2 and 3, we know that the proposed MS-CAE is divided into two parts, namely, the encoder network and the decoder network. Furthermore, the scale of the designed encoder network is relatively small, and the decoding network is more complex than the encoder network. The purpose of this design is to consider the resource limitations of an image monitoring node for the WSN in remote areas. We train the novel MS-CAE network model and extract the weight parameters of the whole model after several periodic iterations. The weight parameters of the well-trained model are divided into two parts, the weight parameters of the encoding network and the decoding network. For the practical deployment of image monitoring for WSN, we require the proposed MS-CAE model to be segmented. The encoder and decoder networks in MS-CAE are deployed to the edge device and cloud-computing device, respectively.

The divided weight parameters are then loaded into the edge device's encoding network and the cloud-computing device's decoding network. For remote monitoring, an edge device is used to collect and compress image data from sensor nodes, which are based on resource-constrained microcontrollers. A cloud-computing device usually has strong computing capability and large storage capacity. Thus, a cloud-computing device is used to parse and restore a large number of reconstructed images.

Therefore, in order to reduce the burden of the edge device in WSN, the relatively small-scale encoding network model parameters are deployed to the edge device. In addition, to improve the quality of the reconstructed image, the weight parameters of the more complex decoding network model are deployed to the cloud device.

5. Experiment Result

5.1. Dataset. Considering the deployment work of the edge device in WSN, for our experiments, we chose a relatively



(a) Zero-padding (b) Reflection-padding

Figure 4: Two kinds of padding mode.

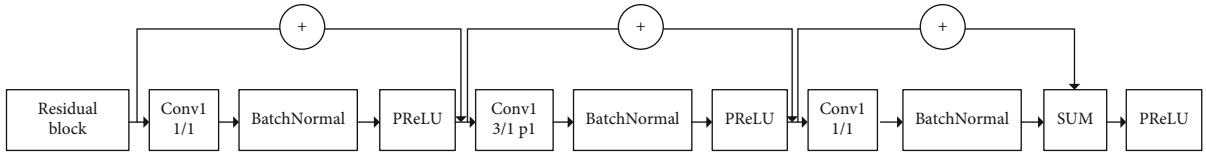


Figure 5: Residual block network.

small image dataset (yt_small_720p) to train and evaluate the performance of the proposed MS-CAE. The dataset covers seven categories: portrait, cartoon, game, natural scenery, advertisement pattern, city scene, and medical image. Furthermore, it collects 2285 images with a resolution of 1280×720. According to the above introduction of pixel block division, we train the proposed MS-CAE network using 60 pixel blocks for each image. In the testing process, we use the Kodak 720p dataset with high-resolution photographs. All procedures are implemented in PyTorch. Each model is trained for 143 epochs on the NVIDIA GeForce RTX 2070 with Max-Q Design GPU.

5.2. Evaluation Indicators. To verify the effectiveness of the proposed MS-CAE, we study the performance with respect to mean square error (MSE), average loss, peak signal-to-noise ratio (PSNR), and structural similarity index measurement (SSIM) for reconstructed compressed image quality. These evaluation indicators are written as follows:

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{m} w_i(y_i - \widehat{y}_i), \tag{16}$$

$$\mathrm{Avgloss}_{\mathrm{per\ patch}} = \frac{1}{60}\sum_{i}^{60} \mathrm{MSEloss}_{\mathrm{per\ patch}}, \tag{17}$$

$$\mathrm{PSNR} = 10 \times \log_{10}\left(\frac{(2^n - 1)^2}{\mathrm{MSE}}\right), \tag{18}$$

$$\mathrm{SSIM}(x, y) = \frac{\left(2\mu_x\mu_y + c_1\right)\left(2\sigma_{xy} + c_2\right)}{\left(\mu_x^2 + \mu_y^2 + c_1\right)\left(\sigma_x^2 + \sigma_y^2 + c_2\right)}, \tag{19}$$

where n is the number of samples and y_i and $\widehat{y}_i$ are the real value and predict value, respectively. In (19), μ_x and μ_y refer to the average values of x and y, respectively. Accordingly, the $\sigma 2x$ is the variance of x and $\sigma 2y$ is the variance of y. σ_{xy} is the covariance of x and y. The $c_1 = (\mathrm{k}_1 L)^2$ and $c_2 = (\mathrm{k}_2 L)^2$ are constants used to maintain stability, where L is the dynamic range of the pixel value and $k_1 = 0.01$ and $k_2 = 0.03$.

5.3. Results

5.3.1. Evaluation for Average Loss Rates. The mean square error is calculated by (16) to measure the error between the real coefficients and the reconstructed coefficients. Average loss reflects the difference in loss between the original image and the reconstructed compressed image. Then, the average loss of the whole image is evaluated by calculating the average loss of 60 pixel blocks by (17). The training loss of a single image can be estimated by averaging the loss of 60 pixel blocks. In Figure 6, we present the average loss of each pixel block between the MS-CAE and CAE models in training over 143 epochs. As shown in Figure 6, the average loss of MS-CAE gradually stabilized and was less than CAE after 80 training epochs. Namely, the training effect of each pixel block of our proposed MS-CAE is better than that of CAE. Moreover, Figure 7 shows the comparison result for the average loss for 24 Kodak images in the test dataset. The result in Figure 7 shows that the average loss of the proposed MS-CAE is obviously lower than that of the CAE.

5.3.2. Quality Evaluation of Reconstructed Images. According to the indicators of PSNR and SSIM in (18) and (19), we evaluate the quality of the reconstructed compressed image for our proposed MS-CAE by setting different bits per pixel (bpp). PSNR is a comprehensive, objective image evaluation indicator that is based on the difference between the corresponding pixels. SSIM focuses on full-reference image quality, which evaluates image similarity based on luminance, contrast, and structure. As a result, these two indicators evaluate the quality of reconstructed compressed images from different perspectives, with higher indexes indicating less distortion. Furthermore, we compute the average PSNR and SSIM values for 24 Kodak images to validate the

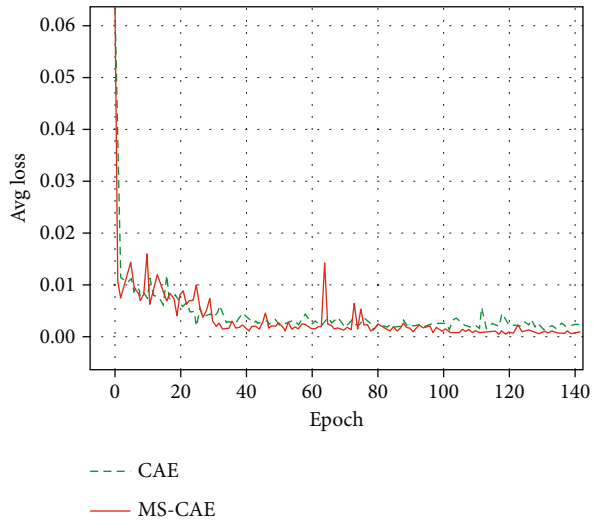


Figure 6: The average training loss of yt_small_720p dataset.

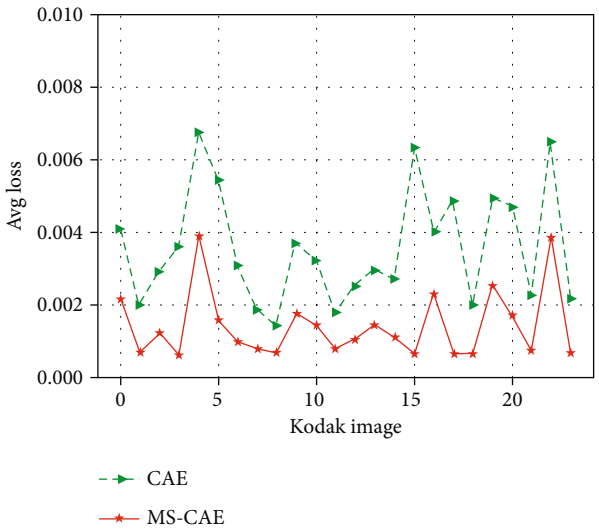


Figure 7: The average testing loss of 24 Kodak images.

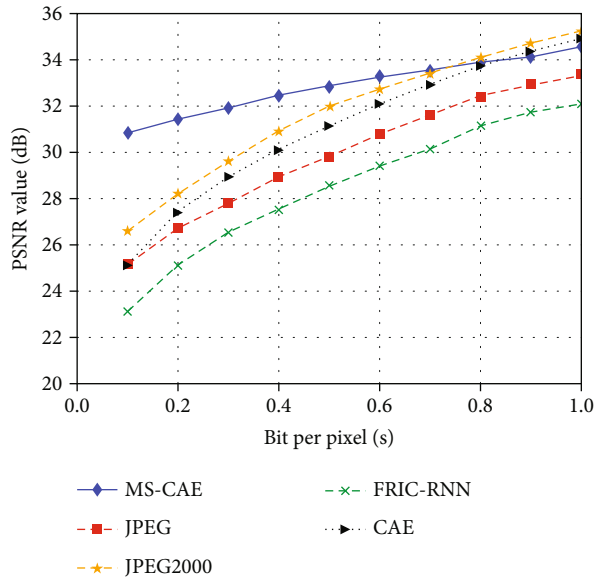


Figure 8: PSNR comparison for different bpp.

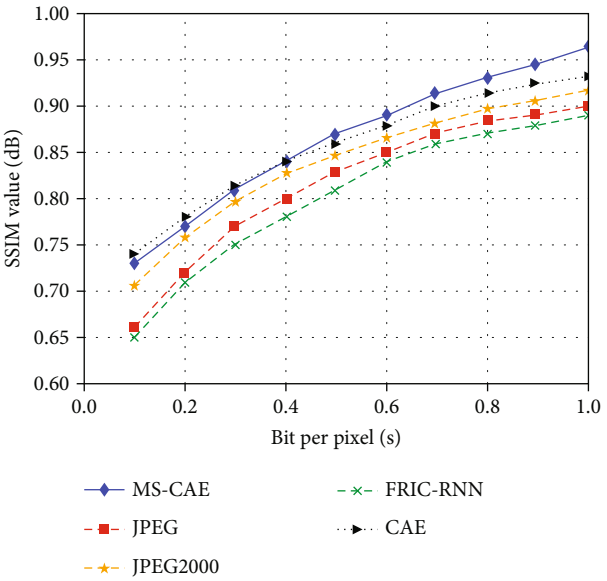


Figure 9: SSIM comparison for different bpp.

performance of five algorithms. The bpp represents the ratio of the number of valid bits in a compressed image to the total number of pixels. Thus, the image compression ratio can also be reflected by the bpp value. The higher bpp value represents a lower image compression ratio and vice versa.

To further verify the performance of the reconstructed compressed image, we compare MS-CAE with JPEG, JPEG 2000, CAE, and Toderici's Full-Resolution Image Compression with Recurrent Neural Networks (FRIC-RNN) [19]. Figure 8 depicts the PSNR comparison value of reconstructed images at various bpp. It can be seen that the PSNR values of MS-CAE are significantly higher than those of JPEG and FRIC-RNN image compression algorithms. Between 0.1042 and ~0.7083 bpp, the reconstructed image quality of MS-CAE is better than that of CAE and JPEG 2000. The results also show that the proposed MS-CAE outperforms other algorithms in terms of high compression ratio. Although the PSNR of MS-CAE is slightly lower than that of CAE and JPEG 2000 in the range of 0.7083~1.0 bpp, the PSNR performance in the range of 0.7083~1.0 bpp which represents a low compression ratio is not a cause for concern for WSN. Furthermore, Figure 9 illustrates the SSIM values of reconstructed images at different bpp. Figure 9 shows that the proposed MS-CAE's structural similarity is greatly improved in the range of 0~1.0 bpp and only slightly lower than that of CAE in the range of 0~0.4 bpp. The reason is that since the proposed MS-CAE algorithm adopts residual block network iterations in the decoding process, it can reduce the network generalization to a small range and is conducive to the feature extraction of high-correlation coefficients.

(a) CAE (PSNR: 28.67 dB, SSIM: 0.8301)

(b) JPEG (PSNR: 28.35 dB, SSIM: 0.76)

(c) MS-CAE (PSNR: 29.94 dB, SSIM: 0.8718)

(d) JPEG 2000 (PSNR: 30.15 dB, SSIM: 0.8215)

Figure 10: Visual effect comparison of Kodak image at 0.3125 bpp.

Therefore, from the above results, our proposed MS-CAE not only improves the decoding network performance of the reconstructed compressed image but also achieves the low-complexity requirement of the encoding network for energy-limited WSN deployment in remote areas.

5.3.3. Visual Effect of Reconstructed Image. In this section, we present the comparison of visual effects between MS-CAE and CAE, JPEG, and JPEG 2000 at 0.3125 bpp for reconstructed images using the Kodak image dataset. The overall comparison results are shown in Figure 10. It can be seen from Figure 10 that the visual effect of the proposed MS-CAE algorithm is the best at a low 0.3125 bpp. This is because the MS-CAE effectively avoids the boundary-blurring effect through reflection-padding. The visual effect of the JPEG algorithm has severe image information distortion. The reason for the phenomenon is that the JPEG algorithm uses an 8×8 matrix of the Discrete Cosine Transform (DCT) to produce a boundary-blurring effect when the pixel blocks are spliced. The original image is compressed by the traditional CAE architecture in Figure 10(a). We can clearly see that the chroma and pixels of the reconstructed compressed image are severely distorted at 0.3125 bpp. We use the JPEG algorithm to compress and reconstruct the same image, as shown in Figure 10(b). Clearly, the PSNR of the reconstructed image in Figure 10(b) is higher than that of the CAE in Figure 10(a). However, because of the boundary-blurring effect caused by the DCT, the SSIM value of JPEG in Figure 10(b) is slightly lower than that of CAE in Figure 10(a). The visual effect of the reconstructed image for JPEG is similar to that of CAE, as shown in Figures 10(c) and 10(d). JPEG 2000's vision effects are also comparable to the proposed MS-CAE. This is because the overall vision effect of the JPEG 2000 algorithm improves significantly as a result of the algorithm's use of the preprocessing procedure, coding, and quantization mode. Furthermore, by utilizing the residual block network and sufficient train epochs, the proposed MS-CAE algorithm avoids block effects and maintains detail elements.

In order to further verify the performance of restoring the image detail texture part, we take the character image in the Kodak image dataset as an example, and the comparison results are shown in Figure 11. Figures 11(a)–11(d) depict the visual effects of CAE, JPEG, MS-CAE, and JPEG 2000, respectively. Figures 11(a) and 11(b) show that the reconstructed image details are not very clear. Their effects in Figures 11(a) and 11(b) are worse than those of MS-CAE and JPEG 2000 in Figures 11(c) and 11(d). From Figures 11(a)–11(d), we know that the proposed MS-CAE algorithm is much clearer in terms of eyelash and hair texture than CAE, JPEG, and JPEG 2000 and has a much higher SSIM value than other algorithms while still maintaining good PSNR performance.

5.3.4. Complexity Analysis of Algorithm. We know from the above sections that our proposed MS-CAE clearly distinguishes itself from the traditional symmetric CAE architecture, and the corresponding encoder and decoder networks

Figure 11: Visual effect comparison of Kodak image details at 0.3125 bpp.

are asymmetric architectures. The purpose of designing the asymmetric architecture is to reduce the parameters of the encoder network for the deployment of an edge device in WSN and to utilize the resource advantages of a cloud-computing device. The encoder network of the proposed MS-CAE reduces the number of network layers, channels, and feature iterations and further improves the computation complexity of image compression. Then, the decoder utilizes three layers of a small residual block network to solve the problem of parameter redundancy and insufficient analytical accuracy so that the quality of the reconstructed image is improved. To analyze the computing complexity of the proposed MS-CAE, we evaluate the average running time of the above-mentioned algorithms in the same experimental environment. The results are shown in Table 2.

As shown in Table 2, the average running time of encoding an image with the proposed MS-CAE is shorter than that of JPEG and JPEG 2000 when using the same computing resource. Although our proposed MS-CAE algorithm consumes slightly more than CAE in the time of single image compression, the accuracy of the reconstructed image is better than that of JPEG and JPEG 2000 at low bpp. This consequence results from many operations of the JPEG and JPEG 2000 image compression algorithms, such as brightness matrix quantization, Huffman coding, DCT, or discrete wavelet transform (DWT). The computation complexity of these operations is high.

Table 2: The average running time of encoding an image for different image compression algorithms.

Algorithm	Average time (s)
JPEG	1.25
JPEG 2000	26
CAE	0.56
MS-CAE	0.67

6. Conclusions

In this paper, a model segmentation-based compressive autoencoder (MS-CAE) image compression algorithm for image monitoring of WSN in remote areas is proposed. We first present the MS-CAE network architecture, which considers the limited computing resources and the practical deployment of WSN. Then, we provide the implementation method for the MS-CAE network. The decoder with a residual block network optimizes the problem of vanishing gradient and gradient exploration. Finally, we split the trained network model and deploy the weight parameters of the encoder and decoder into the edge device and cloud-computing device, respectively. Moreover, for the purpose of obtaining a high-resolution reconstructed compressed image, we appropriately increase the complexity of the decoding network. In addition, we also compare the performance with JPEG, JPEG 2000, FRIC-RNN, and CAE algorithms between 0 and ~1 bpp. The experimental results show that the MS-CAE improves image resolution, compression performance, and transmission efficiency. Based on model segmentation, the designed model MS-CAE has achieved excellent performance in resource savings for edge hardware devices. It also has the ability to completely express the image content. Therefore, it also indicates that the proposed approach effectively improves the monitoring efficiency of long-term environmental image monitoring for WSN.

Data Availability

The image dataset (yt_small_720p) used to support the findings of this study are available from: https://drive.google.com/file/d/1wbwkpz38stSFMwgEKhoDCQCMiLLFVC4T/view.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research was supported by the National Natural Science Foundation of China (Grant Nos. 61961026 and 61962036); Natural Science Foundation of Jiangxi Province, China (Grant No. 20202BABL202003); China Postdoctoral Science Foundation (Grant No. 2020M671556), and Major Science and Technology Projects in Jiangxi Province (Grant No. 20213AAG01012).

References

[1] X. Niu, X. Huang, Z. Zhao, Y. Zhang, C. Huang, and L. Cui, "The design and evaluation of a wireless sensor network for mine safety monitoring," in *IEEE GLOBECOM 2007-2007 IEEE Global Telecommunications Conference*, vol. 3, pp. 1291–1295, Washington, DC, USA, 2007.

[2] M. F. Othman and K. Shazali, "Wireless sensor network applications: a study in environment monitoring system," *Procedia Engineering*, vol. 41, no. 1, pp. 1204–1210, 2012.

[3] A. T. Demetillo, M. V. Japitana, and E. B. Taboada, "A system for monitoring water quality in a large aquatic area using wireless sensor network technology," *Sustainable Environment Research*, vol. 29, no. 1, pp. 1–9, 2019.

[4] X. Bao, L. Han, X. He, W. Tan, and T. Fan, "Optimizing maximum monitoring frequency and guaranteeing target coverage and connectivity in energy harvesting wireless sensor networks," *Mobile Information Systems*, vol. 2019, Article ID 6312589, 14 pages, 2019.

[5] Y. Cao, R. Ji, L. Ji, G. Lei, H. Wang, and X. Shao, "l^2-MPTCP: a learning-driven latency-aware multipath transport scheme for industrial internet applications," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 12, pp. 8456–8466, 2022.

[6] Y. Cao, R. Ji, X. Huang, G. Lei, X. Shao, and I. You, "Empirical mode decomposition-empowered network traffic anomaly detection for secure multipath TCP communications," *Mobile Networks and Applications*, vol. 27, no. 6, pp. 2254–2263, 2022.

[7] J. Ma, M. Li, and H.-J. Li, "Traffic dynamics on multilayer networks with different speeds," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 69, no. 3, pp. 1697–1701, 2022.

[8] J. Peak, J. Hicks, S. Coe, and R. Govindan, "Image-based environmental monitoring sensor application using an embedded wireless sensor network," *Sensors*, vol. 14, no. 9, pp. 15981–16002, 2014.

[9] G. K. Wallace, "The JPEG still picture compression standard," *IEEE Transactions on Consumer Electronics*, vol. 38, no. 1, pp. 43–59, 1991.

[10] M. Rabbani and R. Joshi, "An overview of the JPEG2000 still image compression standard," *Signal Processing: Image Communication*, vol. 17, no. 1, pp. 3–48, 2002.

[11] J. Balle, V. Laparra, and E. P. Simoncelli, "End-to-end optimization of nonlinear transform codes for perceptual quality," in *2016 Picture Coding Symposium (PCS)*, pp. 1–5, Nuremberg, Germany, 2016.

[12] F. Jiang, W. Tao, S. Liu, J. Ren, X. Guo, and D. Zhao, "An end-to-end compression framework based on convolutional neural networks," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 10, pp. 3007–3018, 2017.

[13] P. Kuang, T. Ma, Z. Chen, and F. Li, "Image super-resolution with densely connected convolutional networks," *Applied Intelligence*, vol. 49, no. 1, pp. 125–136, 2019.

[14] C. Huang, H. Liu, T. Chen, Q. Shen, and Z. Ma, "Extreme image coding via multiscale autoencoders with generative adversarial optimization," in *2019 IEEE Visual Communications and Image Processing (VCIP)*, pp. 1–4, Sydney, NSW, Australia, 2019.

[15] Z. Cheng, H. Sun, M. Takeuchi, and J. Katto, "Deep convolutional autoencoder-based lossy image compression," in *2018 Picture Coding Symposium (PCS)*, pp. 253–257, San Francisco, CA, USA, 2018.

[16] L. Theis, W. Shi, A. Cunningham, and F. Huszar, "Lossy image compression with compressive autoencoders," in *International Conference on Learning Representations(ICLR)*, pp. 1–19, Palais des Congrès Neptune, Toulon, France, 2017.

[17] J. Balle, V. Laparra, and E. P. Simoncelli, "End-to-end optimized image compression," in *International Conference on Learning Representations(ICLR)*, pp. 1–27, Palais des Congrès Neptune, Toulon, France, 2017.

[18] Y. Chong, L. Chen, and S. Pan, "End-to-end joint spectral–spatial compression and reconstruction of hyperspectral images

using a 3D convolutional autoencoder," *Journal of Electronic Imaging*, vol. 30, no. 4, 2021.

[19] G. Toderici, D. Vincent, N. Johnston et al., "Full resolution image compression with recurrent neural networks," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5435–5443, Honolulu, HI, USA, 2017.

[20] M. Li, W. Zuo, S. Gu, D. Zhao, and D. Zhang, "Learning convolutional networks for content-weighted image compression," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3214–3223, Salt Lake City, UT, USA, 2018.

[21] M. Akbari, J. Liang, and J. Han, "DSSLIC: deep semantic segmentation-based layered image compression," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 2042–2046, Brighton, UK, 2019.

[22] B. Sushma and B. Fatimah, "Wyner-Ziv coding of chroma in wireless capsule endoscopy image compression using deep side information generation," in *2020 International Conference on Wireless Communications Signal Processing and Networking (WiSPNET)*, pp. 58–62, Chennai, India, 2020.

[23] W. Shi, J. Caballero, F. Huszar et al., "Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1874–1883, Las Vegas, NV, USA, 2016.

[24] Z. Cheng, H. Sun, M. Takeuchi, and J. Katto, "Energy compaction-based image compression using convolutional autoencoder," *IEEE Transactions on Multimedia*, vol. 22, no. 4, pp. 860–873, 2019.

[25] U. Drolia, K. Guo, J. Tan, R. Gandhi, and P. Narasimhan, "Cachier: edge-caching for recognition applications," in *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, pp. 276–286, Atlanta, GA, USA, 2017.

[26] Y. Kang, J. Hauswald, C. Gao et al., "Neurosurgeon," *ACM SIGARCH Computer Architecture News*, vol. 45, no. 1, pp. 615–629, 2017.

[27] C. Ding, A. Zhou, Y. Liu, R. N. Chang, C.-H. Hsu, and S. Wang, "A cloud-edge collaboration framework for cognitive service," *IEEE Transactions on Cloud Computing*, vol. 1, no. 1, p. 99, 2020.

[28] G. Li, J. Wang, H.-W. Shen, K. Chen, G. Shan, and Z. Lu, "CNNPruner: pruning convolutional neural networks with visual analytics," *IEEE Transactions on Visualization and Computer Graphics*, vol. 27, no. 2, pp. 1364–1373, 2020.

[29] S. Roy, P. Panda, G. Srinivasan, and A. Raghunathan, "Pruning filters while training for efficiently optimizing deep learning networks," in *2020 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–7, Glasgow, UK, 2020.

[30] M. Tian and M. Tong, "Self-adaptive subgradient extragradient method with inertial modification for solving monotone variational inequality problems and quasi-nonexpansive fixed point problems," *Journal of Inequalities and Applications*, vol. 2019, no. 1, p. 19, 2019.

[31] Z. Izzo, L. Ying, and J. Zou, "How to learn when data reacts to your model: performative gradient descent," *International Conference on Machine Learning*, vol. 139, pp. 4641–4650, 2021.

[32] A. Jafar and L. Myungho, "Hyperparameter optimization for deep residual learning in image classification," in *2020 IEEE International Conference on Autonomic Computing and Self-Organizing Systems Companion (ACSOS-C)*, pp. 24–29, Washington, DC, USA, 2020.

[33] E. Agustsson, F. Mentzer, M. Tschannen et al., "Soft-to-hard vector quantization for end-to-end learning compressible representations," *Advances in Neural Information Processing Systems*, vol. 2017, p. 30, 2017.

Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 8777890, 10 pages
https://doi.org/10.1155/2022/8777890

Research Article

MEC-Based Cooperative Multimedia Caching Mechanism for the Internet of Vehicles

Longzhe Han,[1] **Sheng Li**,[1] **Chenchen Ao**,[1] **Yan Liu**,[2] **Guangming Liu**,[1] **Yiying Zhang**,[3] **and Jia Zhao**[1]

[1]*School of Information Engineering, Nanchang Institute of Technology, Nanchang 330099, China*
[2]*College of Computer and Software Engineering, East China Normal University, Shanghai 200062, China*
[3]*College of Artificial Intelligence, Tianjin University of Science and Technology, Tianjin 300222, China*

Correspondence should be addressed to Jia Zhao; zhaojia925@163.com

Received 2 May 2022; Accepted 20 August 2022; Published 10 September 2022

Academic Editor: Yuanlong Cao

Multimedia applications are expected to widely deploy over vehicular networks. In order to meet the low-latency and high-speed transmission requirements of multimedia applications, edge caching is introduced to reduce the network traffic and the transmission delay. Due to the limited storage of the edge cache server, an efficient approach for the content management plays a decisive role for the edge cache performance. This paper proposes a vehicle-to-infrastructure-based cooperative caching mechanism for Internet of Vehicles to improve the edge cache utilization. The system model is established with the goal of maximizing the cooperative caching hit rate. To jointly consider the collaborations between macrobase stations (MBS) and multiple roadside units (RSU), we propose a reinforcement learning algorithm to adaptively control the cache management. According to the content popularity and the network status, the proposed algorithm can dynamically adjust cached content across relevant MBSs and RSUs. The simulation results show that the proposed cooperative caching mechanism significantly improve the cache utilization and the quality of services.

1. Introduction

With the rapid development of the Internet of Vehicles (IoV) and 5G communication technology, a large number of multimedia applications, such as traffic video processing, in-vehicle infotainment, and transportation environment monitoring, are emerged to enrich the intelligent transport system [1–3]. To provide high quality of services for multimedia applications in IoV, mobile edge computing (MEC) has attracted the attention as an emerging technology to improve system performance, resource utilization, and reduced transmission delay [4–6]. By introducing computation and storage capabilities to network edge nodes, such as roadside units (RSU) and base stations, the transmission pressure on the core network can be effectively relieved, and at the same time, the content transmission delay can be reduced [7–10]. However, the limited cache space of edge nodes, the time-varying content popularity, the high speed of vehicles, and the constant change of the IoV topology are challenging for the edge cache performance. It is necessary to design an efficient management strategy to efficiently manage the edge cache [11–13].

Currently, there are numerous studies conducted on the subject of edge cache management for IoV. Huang et al. [14] proposed a cache location selection mechanism based on the vehicle trajectory, which can effectively reduce the system load and cache energy consumption. Shi et al. [15] proposed a deep learning communication model based on multimodel compression, which exploited the redundancy between deep learning models in different scenarios to accelerate content transmission in edge networks. In [16], a mixed integer nonlinear programming method was proposed to minimize the cooperative delay between edge servers, and the Lyapunov optimization method was used to optimize the delay problem. In [17], the authors comprehensively considered the mobility of vehicles and proposed an edge caching scheme with perceptible mobility probability. By dividing the data

into data blocks of different sizes and buffering these data blocks in the edge server close to the vehicle, the overhead and transmission delay of backhaul traffic were reduced. Meng et al. [18] studied the cache service strategy of offline networking in the edge computing environment and proposed a cache storage algorithm on node core. In [19], a new information-centric heterogeneous network framework was designed, using a distributed algorithm with alternating-direction multipliers to solve the problem of cache resource allocation. In order to further reduce the transmission delay and improve the response rate, the authors [20] proposed a cooperative cache allocation and calculation offload scheme, and the MEC servers were cooperated to perform calculation tasks and data caching. With the rapid development of artificial intelligence, the deep reinforcement learning [21] has been widely used in edge caching and resource allocation of vehicle networks with its unique perception and decision-making capabilities [22, 23].

The main contributions of this paper are as follows: to take full advantage of edge cache resources, a hierarchical cooperative architecture, including MBSs, RSUs, and vehicles, are introduced. We establish a Markov decision model based on the proposed architecture to describe the cooperative caching process. We propose a reinforcement learning cache management algorithm, which follows the Deep Deterministic Policy Gradient (DDPG) scheme. The proposed algorithm has fast convergence rate and can self-adapt to the complex network environment.

The rest of this paper is organized as follows. Section 2 presents the system model for cooperative edge caching. Section 3 discusses the proposed cooperative caching mechanism. The experiment settings and result analysis are presented in Section 4. Finally, in the Section 5, we discuss concluding remarks and our future work.

2. System Model

2.1. Cooperative Edge Caching Model. In order to make full use of the storage of MBSs, RSUs, and vehicles, we construct a three-layer cooperative cache architecture, as shown in Figure 1. The core layer includes MBSs and the backhaul network, and the MBSs are connected to the RSUs through wired links. For the cooperative RSU layer, it consists of RSUs distributed in different areas, and the RSUs communicate through wireless links. The vehicle layer includes vehicles running in different areas. MBSs, RSUs, and vehicles have storage to temporarily buffer certain amount of content. Initially, the vehicle sends a content request. If the vehicle itself has the content, it will obtain directly from its cache. If not, it will send the request to the local RSU. If the local RSU does not store the content, the local RSU queries the cooperative RSUs. If neither the local RSU nor the cooperative RSUs have the content, the request is sent to the core layer.

MBS is responsible for collecting system status information, controlling global resource management, and content caching decisions. Compared with obtaining content from a remote server, the cooperative caching model can effectively reduce the transmission delay and transmission cost. The set of RSUs can be expressed as $R = \{1, 2, 3, \cdots, R\}$. $V = \{v_1, v_2, \cdots, v_N\}$ represents the set of vehicles under the coverage of the RSU. The RSU is responsible for collecting relevant information of vehicles under its own coverage area and uploading to the MBS.

2.2. Content Delivery Model. In the multilevel cooperative edge caching model, the vehicle v_i can send content requests to the RSU or adjacent vehicles. Vehicles within the coverage area of one RSU use the same frequency band to communicate, and it is leading to interference between vehicles. Therefore, the transmission rate from RSU r to vehicle v can be obtained from Shannon's formula as

$$R_{r,v} = b_{r,v} B_R \log\left(1 + \frac{p_r h_{r,v}}{\sigma^2 + \sum_{v'=1,v'\neq v}^{V} p_r h_{r,v'}}\right), \quad (1)$$

where $b_{r,v}$ represents the channel bandwidth allocated by the RSU r to the vehicle v, B_R represents the channel bandwidth of the RSU r, and p_r is the transmission power of the RSU r. $h_{r,v}$ is the channel gain between the RSU r and the vehicle v, and σ^2 represents noise power. $\sum_{v'=1,v'\neq v}^{V} p_r h_{r,v'}$ is the V2I communication downlink interference [24].

Orthogonal frequency division multiple access (OFDMA) is used between MBSs and vehicles. Vehicles associated with the MBS are assigned an orthogonal subcarrier, and the transmission rate from the MBS to vehicle v_i is

$$R_{m,v} = \frac{B_m}{W} \log\left(1 + \frac{p_m h_{m,v}}{\sigma^2}\right), \quad (2)$$

where B_m is the channel bandwidth of the vehicle, and p_m represents the transmission power of the vehicles. $h_{m,v}$ is the channel gain between the vehicle v and the MBS, and σ^2 represents the noise power [25].

2.3. Content Popularity Model. Assuming that there are K content requests, then the request probability of these K contents are $P_1, P_2, P_3, \cdots, P_K$, and the probability obeys the Zip$f$ distribution [26]. The relationship between the content request probability and the content popularity level can be expressed as [27]

$$P(s) = \frac{\Phi}{s^\theta} s \in \{1, 2, 3 \cdots K\}, \quad (3)$$

$$\Phi = \sum_{i=1}^{K} \frac{1}{i^{-\theta}}^{-1}, \quad (4)$$

where s represents the content popularity level, and θ is the Zip$f$ impact factor, also known as the popularity slope. If θ is getting larger, the distribution of Zip$f$ is steeper, and the popularity tends to be concentrated [28, 29]. The value of the Zip$f$ factor depends on the users' behavior. The relationship between the request probability and popularity level can be further expressed as

$$P(s) = \frac{s^{-\theta}}{\sum_{i=1}^{K} i^{-\theta}}. \quad (5)$$

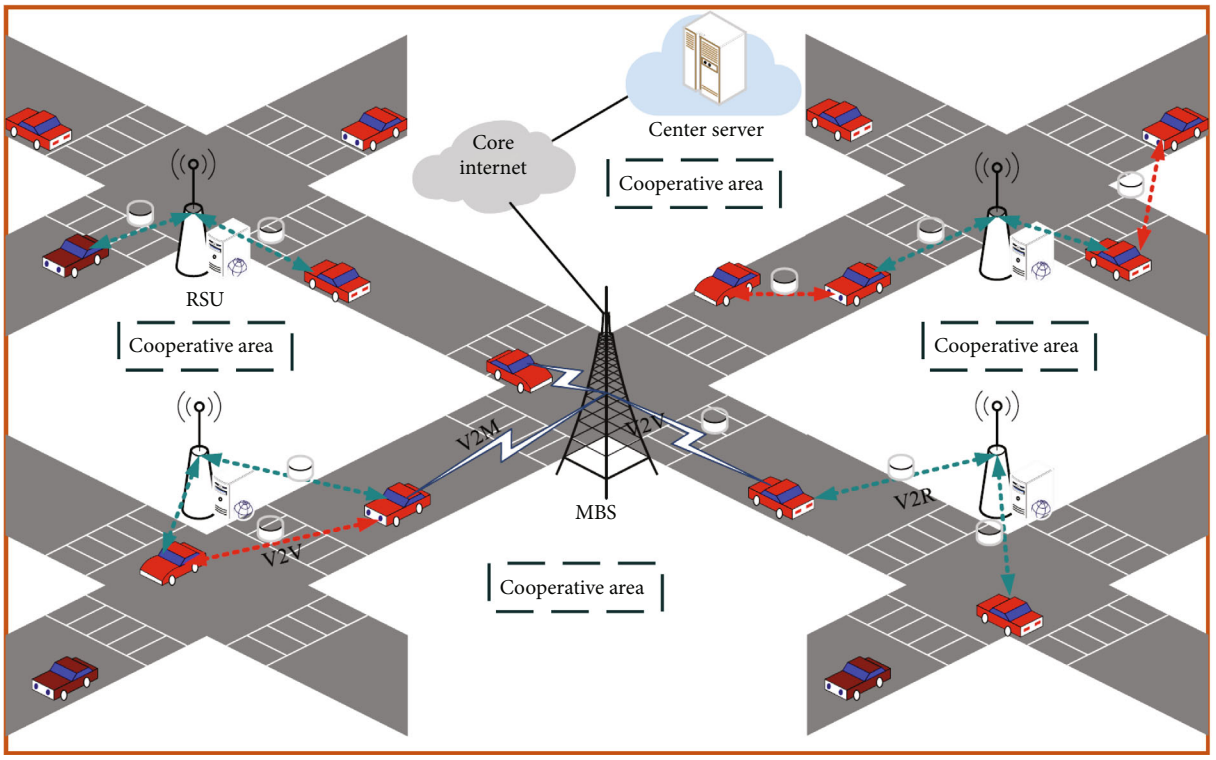


Figure 1: Cooperative edge cache model of IoV.

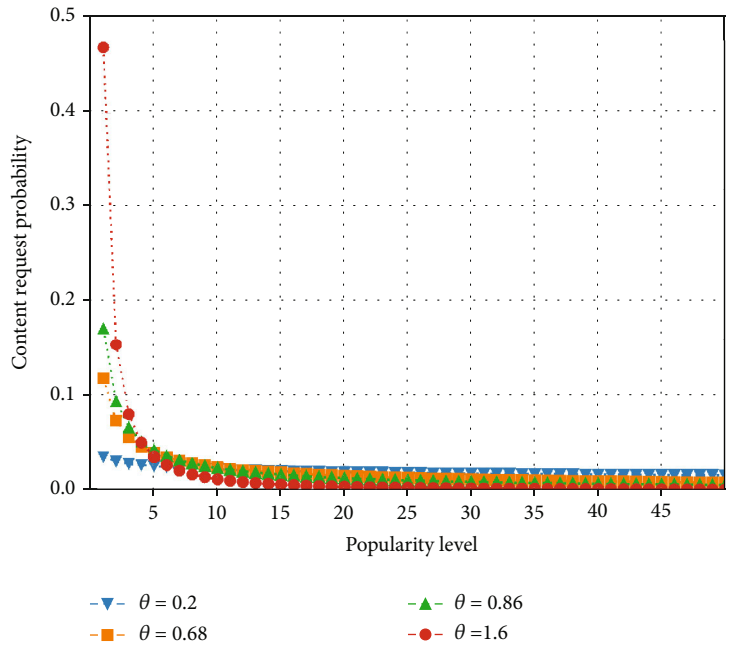


Figure 2: Relationship between content popularity level and request probability.

Figure 2 shows the relationship between the popularity level and the request probability. It can be seen that the influence of the popularity inclination on the request probability distribution. The content with high request probability only accounts for a small part of all content [30].

3. Cooperative Caching Mechanism

3.1. Problem Model. The cooperative caching is able to theoretically achieve a high cache hit rate than the noncooperative caching. We use a binary variable $c_{i,k} \in \{0, 1\}, i \in R, k \in K$ to

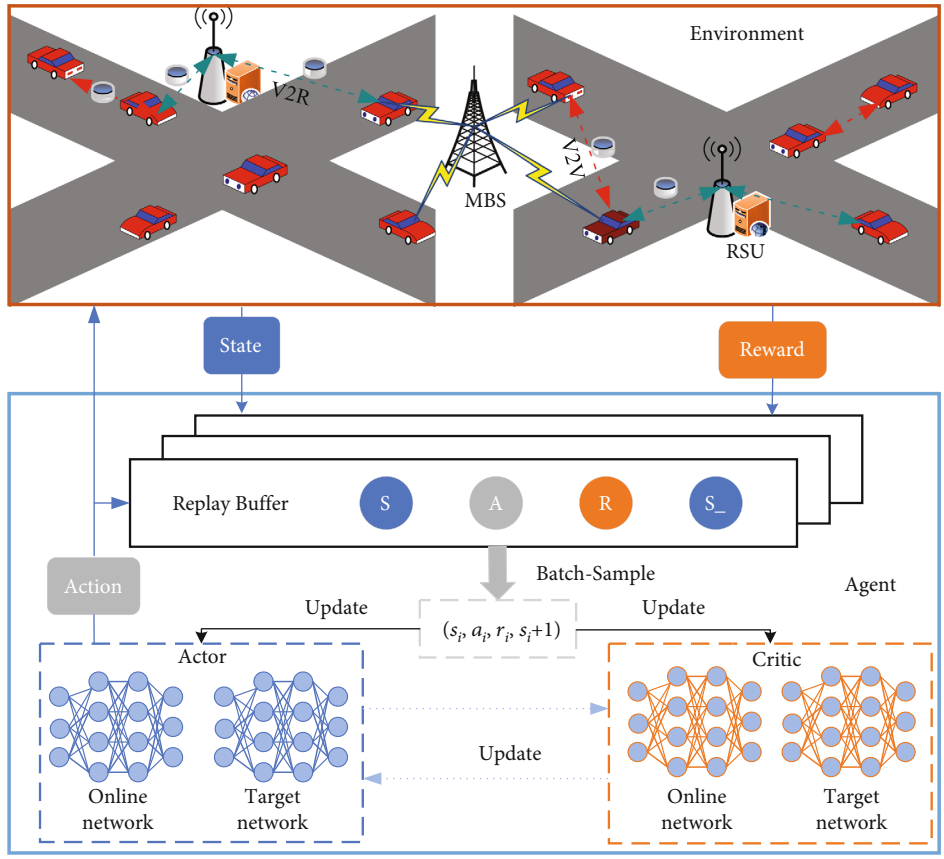


Figure 3: Schematic of the cooperative caching algorithm.

represent the cache state of the content k of RSU_i. $c_{i,k} = 1$ means that the content k is cached in the RSU_i, and $c_{i,k} = 0$ means that the RSU_i does not cache the content k. The cooperative edge cache hit rate of the RSU_i can be expressed as

$$h_i = \sum_{k \in K} p_k \left(c_{i,k} + \sum_{n \in R} c_{n,k} \delta_{n,i} \right), \tag{6}$$

where the binary variable $\delta_{n,i} \in \{0, 1\}$ indicates whether the RSU_n is cooperated with the RSU_i. Therefore, the average cooperative cache hit ratio of the system can be expressed as

$$\bar{h} = \frac{1}{R} \sum_{i \in R} h_i = \frac{1}{R} \sum_{i \in R} \sum_{k \in K} p_k h_i = \frac{1}{R} \sum_{i \in R} \sum_{k \in K} p_k \left(c_{i,k} + \sum_{n \in R} c_{n,k} \delta_{n,i} \right). \tag{7}$$

For RSU_i, the size of the cache space is S^{RSU_i}, and then the optimization problem of maximum average cooperative cache hit rate can be expressed as the following:

$$\begin{aligned} \max_{\{c_{i,k}\}} \quad & \bar{h} \\ s.t \quad & c_{i,k} \in \{0, 1\}, i \in R, k \in K \\ & \sum_{k \in K} c_{i,k} \le S^{RSU^i}, i \in R. \end{aligned} \tag{8}$$

Regarding to the vehicle cache, the cache hit rate of the vehicle j can be expressed as

$$h_j = \sum_{k \in K} p_k c_{j,k}. \tag{9}$$

Therefore, the average cache hit rate for all vehicles is expressed as

$$\bar{h}_j = \frac{1}{V} \sum_{j \in V} h_j = \frac{1}{V} \sum_{j \in V} \sum_{k \in K} p_k h_j = \frac{1}{V} \sum_{j \in V} \sum_{k \in K} p_k c_{j,k}. \tag{10}$$

The size of the cache space of the vehicle j is S^{v^j}. Under the limitation of the cache space, the problem of the maximum average cache hit rate of vehicles can be expressed as the following:

$$\begin{aligned} \max_{\{c_{j,k}\}} \quad & \bar{h}_j \\ s.t \quad & c_{j,k} \in \{0, 1\}, j \in V, k \in K \\ & \sum_{k \in K} c_{j,k} \le S^{v^j}, j \in V. \end{aligned} \tag{11}$$

Maximizing the cache hit rate of the system is to maximize the average cache hit rate of the cooperative caches and vehicles, as the following form:

1: Initialize Actor online network parameters θ^{Q}, Critic online network parameters θ^{μ}, experience replay memory M
2: Initialize Actor target network parameters $\theta^{Q'}$, Critic target network parameters $\theta^{\mu'}$
3: Initialize caching state of RSUs, MBS and Vehicles, content popularity
4: **for** episode =1, M **do**
5: Environment state space initialization, initialization system cache hit rate
6: Randomly choose action N as action exploration
7: **for** t =1,2,3,…,T **do**
8: Select action $a_t = \mu(s_t, \theta^{\mu}) + N_t$ according to observed state s_t and current strategy
9: Calculate reward $R(t)$ based on current selection action a_t and state s_t, update state $s_t \longrightarrow s_{t+1}$
10: Update the reward $R(t) = CP\sum\sum_{v}(\bar{h}(t) + \bar{h}_j(\mathrm{t}))$ and store $(s_t, a_t, R(t), s_{t+1})$ in M
11: Randomly sample N samples from the experience replay memory M, $(s_i, a_i, R(t^i), s_{i+1})$
12: Evaluate $y_i = R(t^i) + \gamma Q'(s_{i+1}, \mu'(s_{i+1}|\theta^{\mu'})|\theta^{Q'})$
13: Update Critic Network Parameters θ^{μ} by Minimizing Loss
14: $L(\theta) = 1/N\sum_i (y_i - Q(s_i, a_i|\theta^{Q}))^2$
15: Update Actor Network Parameters θ^{Q} via Policy Gradients
16: $\nabla_{\theta\mu} J \approx 1/N\sum_i \nabla_a Q(s, a|\theta^{Q})|_{s=s_i, a=\mu(s_i)} \nabla_{\theta\mu}\mu(s, |\theta^{\mu})|_{s_i}$
17: Update target network parameters
18: $\theta^{Q'} \longleftarrow \tau\theta^{Q} + (1-\tau)\theta^{Q'}$
19: $\theta^{\mu'} \longleftarrow \tau\theta^{\mu} + (1-\tau)\theta^{\mu'}$
20: **end for**
21: **end for**

Algorithm 1: Cooperative edge caching algorithm for Internet of Vehicles based on DDPG.

$$\begin{aligned} \max_{\{c_{i\&j,k}\}} \quad & H = \left(\bar{h} + \bar{h}_j\right), \\ s.t \quad & \mathrm{C1}: c_{i,k} \in \{0,1\}, i \in R, k \in K, \\ & \mathrm{C2}: c_{j,k} \in \{0,1\}, j \in V, k \in K, \\ & \mathrm{C3}: \sum_{k \in K} c_{i,k} \le S^{RSU^i}, i \in R, \\ & \mathrm{C4}: \sum_{k \in K} c_{j,k} \le S^{v^j}, j \in V. \end{aligned} \tag{12}$$

3.2. Cooperative Caching Algorithm Based on DDPG. To solve the optimization problem in the previous section, it is necessary to build the Markov decision process for the cooperative edge caching scenario. The Markov decision process is a tuple including state, action, and reward. The components are defined as follows: the system state at each time t is defined as $s_t = [S^{\mathrm{MBS}}, R * S^{\mathrm{RSU}}, V * S^{v}, q_t]$, that is, at time t, the cooperative cache space, cache state information, vehicle cache information, and vehicle content request. The action space at each time t is defined as $a_t = [a_0, a_1, a_2, a_3]$, where a_0 represents the content cached in the MBS, a_1 means the content cached in the RSU, a_2 represents the content cached in the vehicle itself, and a_3 is for the content cached in the randomly.

The system joint reward function $R(t)$ is expressed as $R(t) = CP\sum\sum_{v}(\bar{h}(t) + \bar{h}_j(\mathrm{t}))$, where C is the characteristic constant, and $\bar{h}(t) + \bar{h}_j(\mathrm{t})$ is the average cache hit rate of the cooperative cache and the average cache hit rate of the vehicle j at time t, respectively. P represents the penalty coefficient given by the vehicle.

Table 1: Simulation parameters.

System parameter	Value/description
RSU and MBS cache capacity	10 TB, 15 TB
Vehicle cache capacity	5 TB
RSU and MBS transmission power	35 dBm, 38 dBm
Number of contents	7000, 9000, 10000
Zip$f$ impact factor	0.68
Noise power	-95 dBm
Wireless bandwidth	10 MHz
Wired bandwidth	20 MHz
Number of vehicles	25, 30, 35, 40, 45
Number of neural network layers	2
Number of neurons	[300, 400]
Learning rate	0.00025
Batch sampling size	64
Replay buffer size	7500

The system block diagram of the cooperative edge caching algorithm is shown in Figure 3. The environment consists of an actor network, a critical critic, and an experience replay memory. The actor and critic network are both composed of two different deep neural networks. The online network is used

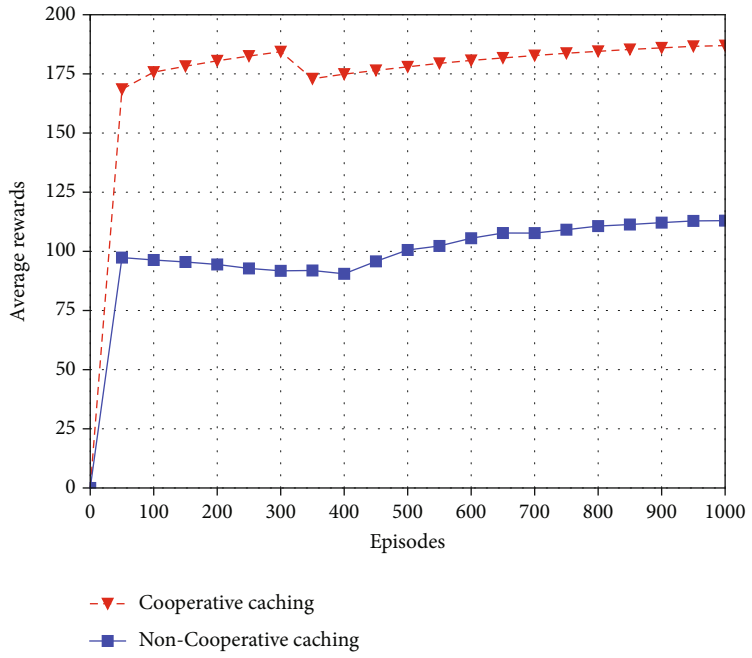


Figure 4: Comparison of cumulative average rewards of different caching schemes.

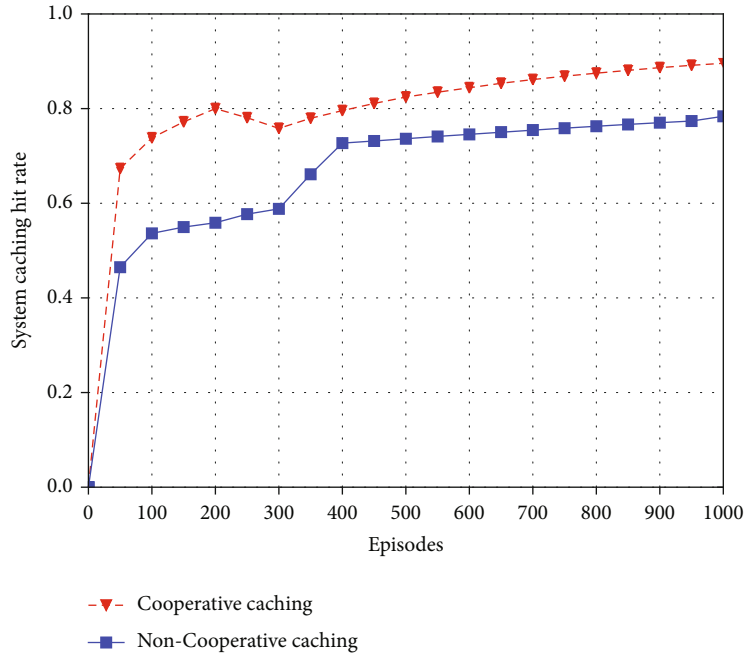


Figure 5: The impact of different caching strategies on the system cache hit rate.

for actions, and the target network is used for the evaluation of actions. The agent receives the environmental state information and executes the corresponding the action. Algorithm 1 shows the flow of the cooperative edge caching algorithm [16]. First, it initializes the network parameters of the actor network, the critic network, and the experience replay memory. After the parameter initialization is completed, the agent obtains the environmental state information and makes a decision for the content caching. The agent receives immediate reward feedback from the system, and the system enters the next new state. The agent store the current status information into the experience replay memory for future training.

4. Experiment Results and Analysis

In the simulation environment, the cache capacity of the MBS is 15 TB, and the coverage radius is 2 km. The RSU

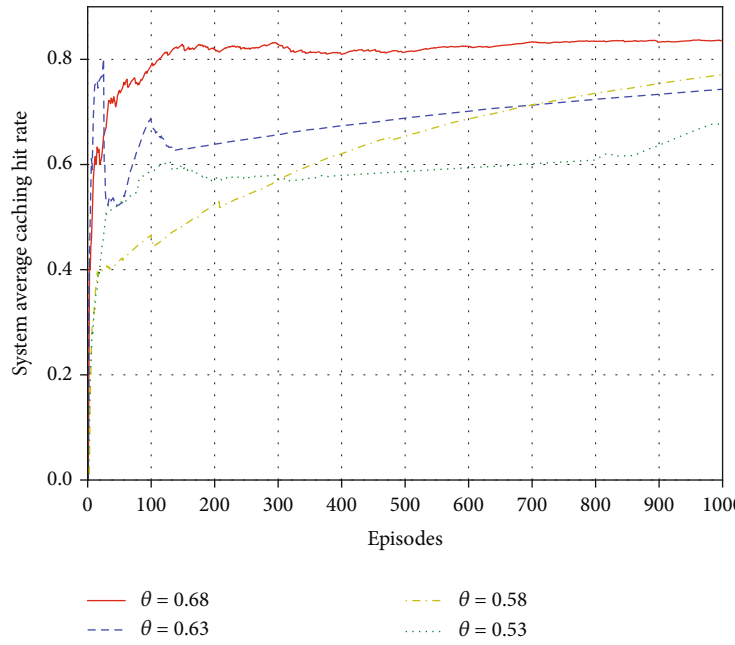


Figure 6: Comparison of the average hit rate under different Zipf distributions.

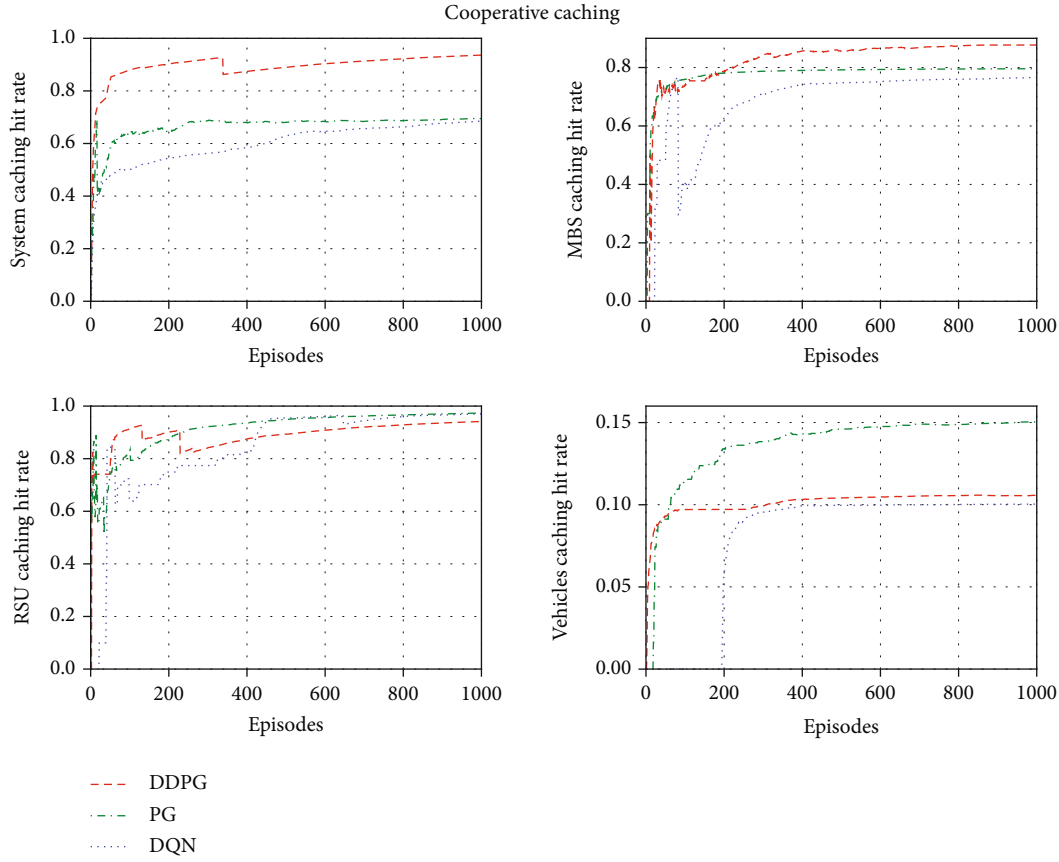


Figure 7: Comparison of the cache hit ratio under the cooperative cache scheme.

cache capacity is 10 TB, and the coverage radius is 200 m. The range of the number of vehicles is from 25 to 45, and the size of the vehicle cache is 5 TB. The transmission power of RSU and MBS content is 35 dBm and 38 dBm, respectively. The Zip*f* impact factor is 0.68. The neural network parameters are set as two hidden layers, and the activation

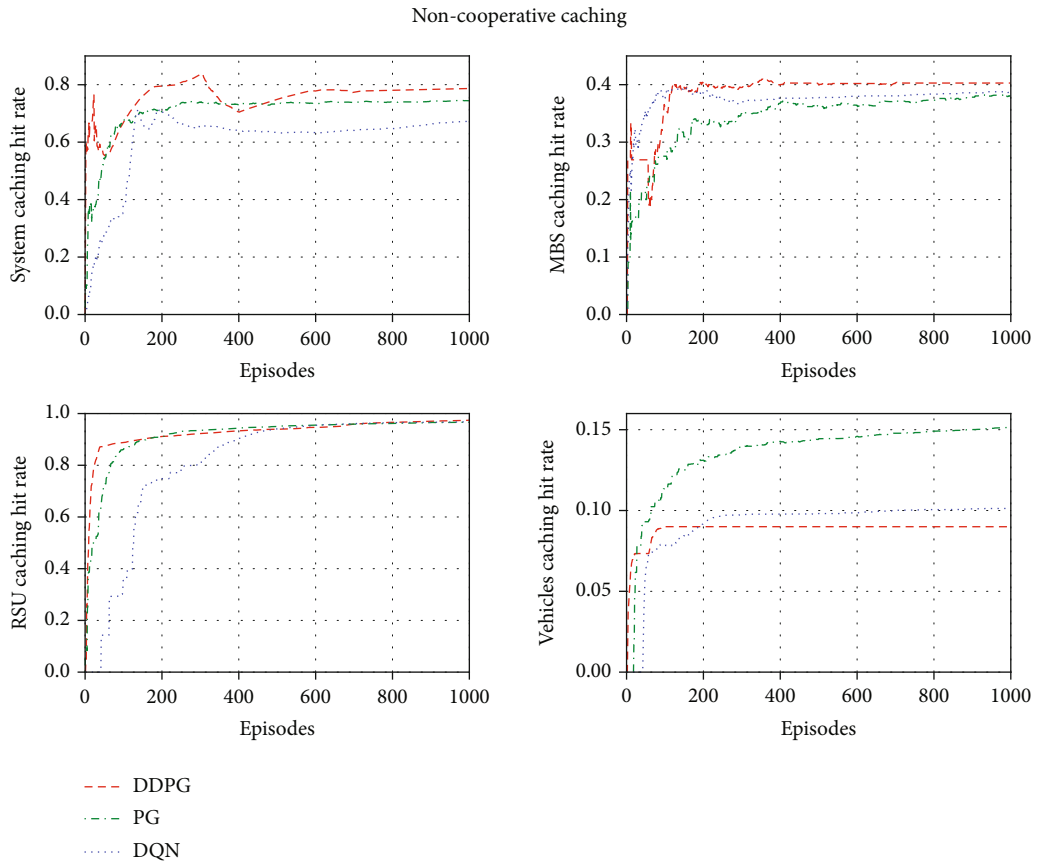


Figure 8: Comparison of cache hit ratios under the noncooperative cache scheme.

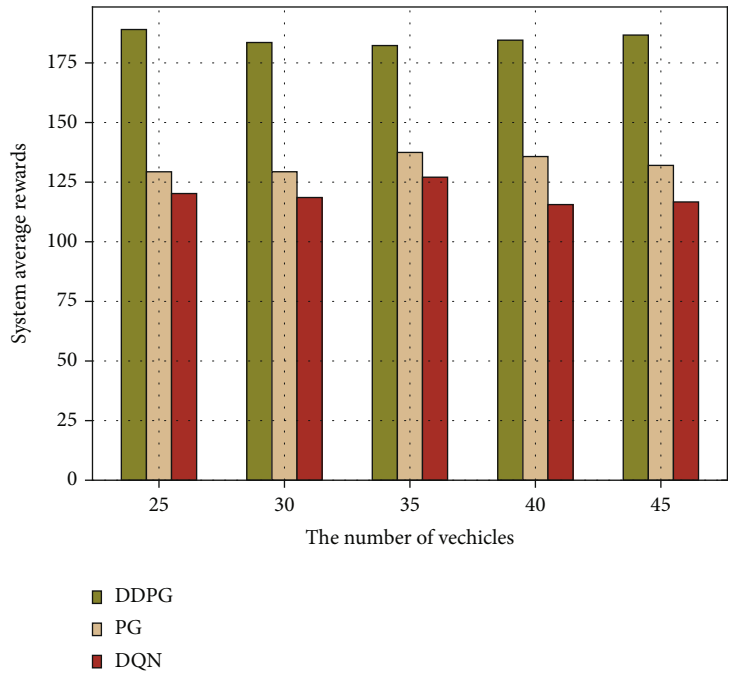


Figure 9: The effect of the number of vehicles under different algorithms.

function is ReLU [15]. The detailed settings of parameters are shown in Table 1.

To verify the performance of the cooperative edge caching strategy, we compare the cooperative caching scheme with noncooperative caching scheme under the parameters setting of Table 1. The learning process of cooperative and noncooperative caching strategies is shown in Figure 4. The average reward of cooperative caching rapidly increases to 170 after 50 episodes and then gradually stabilizes. For noncooperative caching, with the increase of training episodes, the average reward has stabilized around 100. The cooperative caching strategy not only make the most use of the cache space but also effectively improve the system performance. The advantage of noncooperative caching is that it does not need to consider the content caching status of other edge servers, and the system complexity is low.

Figure 5 shows the comparison of the cache hit rate for different caching strategies. With the continuous increase of training times, the cooperative cache hit rate obtained by the system is stable above 85%. For the noncooperative caching, the system cache hit rate is roughly 10% lower than the cooperative caching. When the training reaches 400 rounds, the hit rate of the noncooperative cache gradually tends to 75%. The reason for the gap is that the noncooperative caching cannot make full use of the cache space, which causes a waste of storage and a low system caching hit rate.

The relationships of the system caching hit rate under different Zip$f$ distributions is shown in Figure 6. When the Zip$f$ distribution parameter is large, it indicates that the vehicle users have more requests for the content with high popularity. Caching the high popularity content is beneficial to the improvement of the system cache hit rate. When the content requests increase, the noncooperative caching is difficult to meet the vehicle requests. The proposed cooperative cache strategy fully considers the cooperation between RSU and MBS, and the system cache hit rate increases significantly.

Figures 7 and 8 show the comparison of the average cache hit rate of different algorithms under the different schemes. With the increase of training times, the average caching hit rate of MBS gradually tends to a stable value above 80%. For the (deep Q network) DQN algorithm, the caching hit rate fluctuates greatly in the first 250 episodes, because DQN is difficult to deal with the complex state information. For the average cache hit rate of RSU, the DDPG algorithm has better performance in the first 250 episodes. After 250 training episodes, the effect is slightly lower than that of the (policy gradient) PG and the DQN algorithm.

Figure 9 presents the cooperative cache performance under the different numbers of vehicles. Compared with PG and DQN based algorithms, the DDPG-based algorithm can bring better benefits to the system and tend to be stable when dealing with the complex environment. At the same time, it also verifies that the DDPG-based algorithm has unique advantage for improving the overall average hit rate.

5. Conclusions

This paper focus on the improving of cache performance in the IoV environment and proposes a V2I-based cooperative caching strategy. We propose MBS-RSU-vehicle three layer architecture and model the problem as maximizing the average cooperative cache hit rate. The objective function is solved by using the reinforcement learning algorithm based on DDPG. In order to verify the performance of the proposed cache strategy, the effects of cooperative caching and noncooperative are compared under different system parameters. In future work, we will further consider the content transmission delay and use game theory to solve the problem of resource competition between cooperative cache servers.

Data Availability

The data used to support the study are available within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research is jointly supported by the National Natural Science Foundation of China (No. 61962036, No. 52069014, No. 61961026), by the Science and Technology Research Project of Jiangxi Provincial Department of Education (No. GJJ190957), and by the Scientific Research Foundation for the Returned Overseas Chinese Scholars, State Education Ministry.

References

[1] K. Zhang, S. Leng, Y. He, S. Maharjan, and Y. Zhang, "Cooperative content caching in 5G networks with mobile edge computing," *IEEE Wireless Communications*, vol. 25, no. 3, pp. 80–87, 2018.

[2] X. Xu, H. Li, W. Xu, Z. Liu, L. Yao, and F. Dai, "Artificial intelligence for edge service optimization in Internet of Vehicles: a survey," *Tsinghua Science and Technology*, vol. 27, no. 2, pp. 270–287, 2022.

[3] B. Jedari, G. Premsankar, G. Illahi, M. D. Francesco, A. Mehrabi, and A. Ylä-Jääski, "Video caching, analytics, and delivery at the wireless edge: a survey and future directions," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 1, pp. 431–471, 2021.

[4] Z. Wu and D. Yan, "Deep reinforcement learning-based computation offloading for 5G vehicle-aware multi-access edge computing network," *China Communications*, vol. 18, no. 11, pp. 26–41, 2021.

[5] H. Wu, J. Zhang, Z. Cai, F. Liu, Y. Li, and A. Liu, "Toward energy-aware caching for intelligent connected vehicles," *IEEE Internet of Things Journal*, vol. 7, no. 9, pp. 8157–8166, 2020.

[6] C. Song, W. Xu, T. Wu, S. Yu, P. Zeng, and N. Zhang, "QoE-driven edge caching in vehicle networks based on deep reinforcement learning," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 6, pp. 5286–5295, 2021.

[7] I. A. Elgendy, W. Z. Zhang, H. He, B. B. Gupta, A. El-Latif, and A. Ahmed, "Joint computation offloading and task caching for multi-user and multi-task MEC systems: reinforcement learning-based algorithms," *Wireless Networks*, vol. 27, no. 3, pp. 2023–2038, 2021.

[8] P. Yang, N. Zhang, S. Zhang, L. Yu, J. Zhang, and X. Shen, "Content popularity prediction towards location-aware mobile edge caching," *IEEE Transactions on Multimedia*, vol. 21, no. 4, pp. 915–929, 2019.

[9] B. Mao, F. Tang, Z. M. Fadlullah et al., "A novel non-supervised deep-learning-based network traffic control method for software defined wireless networks," *IEEE Wireless Communications*, vol. 25, no. 4, pp. 74–81, 2018.

[10] M. Chen, Y. Hao, L. Hu, K. Huang, and V. K. N. Lau, "Green and mobility-aware caching in 5G networks," *IEEE Transactions on Wireless Communications*, vol. 16, no. 12, pp. 8347–8361, 2017.

[11] J. Luo, F. R. Yu, Q. Chen, and L. Tang, "Adaptive video streaming with edge caching and video transcoding over software-defined mobile networks: a deep reinforcement learning approach," *IEEE Transactions on Wireless Communications*, vol. 19, no. 3, pp. 1577–1592, 2020.

[12] R. Xie, J. Wu, R. Wang, and T. Huang, "A game theoretic approach for hierarchical caching resource sharing in 5G networks with virtualization," *China Communications*, vol. 16, no. 7, pp. 32–48, 2019.

[13] T. Zhang, X. Xu, L. Zhou, X. Jiang, and J. Loo, "Cache space efficient caching scheme for content-centric mobile ad hoc networks," *IEEE Systems Journal*, vol. 13, no. 1, pp. 530–541, 2019.

[14] X. Huang, K. Xu, Q. Chen, and J. Zhang, "Delay-aware caching in internet-of-vehicles networks," *IEEE Internet of Things Journal*, vol. 8, no. 13, pp. 10911–10921, 2021.

[15] P. Shi, F. Gao, S. Liang, and S. Yu, "Multi-model inference acceleration on embedded multi-core processors," in *Proceedings of 2020 International Conference on Intelligent Computing and Human-Computer Interaction (ICHCI)*, pp. 400–403, Sanya, China, 2020.

[16] L. T. Tan and R. Q. Hu, "Mobility-aware edge caching and computing in vehicle networks: a deep reinforcement learning," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 11, pp. 10190–10203, 2018.

[17] Z. Zhang, J. Dai, M. Zeng, D. Liu, and S. Mao, "Scalable video caching for information centric wireless networks," *IEEE Access*, vol. 8, pp. 77272–77284, 2020.

[18] Z. Meng, Z. Weicheng, L. Yan, and Z. Lidong, "The information center Internet of Vehicles (IoV) cache service strategy based on mobile edge computing," in *Proceedings of 2020 7th International Conference on Information Science and Control Engineering (ICISCE)*, pp. 1443–1448, Changsha, China, 2020.

[19] G. Manogaran, V. Saravanan, and C.-H. Hsu, "Information-centric content management framework for software defined Internet of Vehicles towards application specific services," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 7, pp. 4541–4549, 2021.

[20] X. Peng and H. Garg, "Intuitionistic fuzzy soft decision making method based on CoCoSo and CRITIC for CCN cache placement strategy selection," *Artificial Intelligence Review*, vol. 55, no. 2, pp. 1567–1604, 2022.

[21] S. Xu, X. Liu, S. Guo, X. Qiu, and L. Meng, "MECC: a mobile edge collaborative caching framework empowered by deep reinforcement learning," *IEEE Network*, vol. 35, no. 4, pp. 176–183, 2021.

[22] L. Ale, N. Zhang, H. Wu, D. Chen, and T. Han, "Online proactive caching in mobile edge computing using bidirectional deep recurrent neural network," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 5520–5530, 2019.

[23] L. Wang, H. Wu, Z. Han, P. Zhang, and H. V. Poor, "Multi-hop cooperative caching in social IoT using matching theory," *IEEE Transactions on Wireless Communications*, vol. 17, no. 4, pp. 2127–2145, 2018.

[24] X. Zhang, H. Li, J. Wang et al., "Data-driven caching with users' content preference privacy in information-centric networks," *IEEE Transactions on Wireless Communications*, vol. 20, no. 9, pp. 5744–5753, 2021.

[25] L. Yao, Y. Wang, X. Wang, and W. U. Guowei, "Cooperative caching in vehicular content centric network based on social attributes and mobility," *IEEE Transactions on Mobile Computing*, vol. 20, no. 2, pp. 391–402, 2021.

[26] Z. Chang, L. Lei, Z. Zhou, S. Mao, and T. Ristaniemi, "Learn to cache: machine learning for network edge caching in the big data era," *IEEE Wireless Communications*, vol. 25, no. 3, pp. 28–35, 2018.

[27] X. Wang, Y. Han, C. Wang, Q. Zhao, X. Chen, and M. Chen, "In-edge AI: intelligentizing mobile edge computing, caching and communication by federated learning," *IEEE Network*, vol. 33, no. 5, pp. 156–165, 2019.

[28] J. Zhang and K. B. Letaief, "Mobile edge intelligence and computing for the Internet of Vehicles," *Proceedings of the IEEE*, vol. 108, no. 2, pp. 246–261, 2020.

[29] Y. Qian, Y. Jiang, L. Hu, M. S. Hossain, M. Alrashoud, and M. Al-Hammadi, "Blockchain-based privacy-aware content caching in cognitive Internet of Vehicles," *IEEE Network*, vol. 34, no. 2, pp. 46–51, 2020.

[30] R. Wang, Z. Kan, Y. Cui, D. Wu, and Y. Zhen, "Cooperative caching strategy with content request prediction in Internet of Vehicles," *IEEE Internet of Things Journal*, vol. 8, no. 11, pp. 8964–8975, 2021.

Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 1237619, 17 pages
https://doi.org/10.1155/2022/1237619

Research Article

Dynamic Rendering-Aware VR Service Module Placement Strategy in MEC Networks

Chunyu Liu, Heli Zhang, Xi Li, and Hong Ji

Key Laboratory of Universal Wireless Communications, Ministry of Education, Beijing University of Posts and Telecommunications, Beijing 100876, China

Correspondence should be addressed to Heli Zhang; zhangheli@bupt.edu.cn

Received 10 April 2022; Revised 25 July 2022; Accepted 1 August 2022; Published 18 August 2022

Academic Editor: A.H. Alamoodi

Combining multiaccess edge computing (MEC) technology and wireless virtual reality (VR) game is a promising computing paradigm. Offloading the rendering tasks to the edge node can make up for the lack of computing resources of mobile devices. However, the current offloading works ignored that when rendering is enabled at the MEC server, the rendering operation depends heavily on the environment deployed on this MEC serve. In this paper, we propose a dynamically rendering-aware service module placement scheme for wireless VR games over the MEC networks. In this scheme, the rendering tasks of VR games are offloaded to the MEC server and closely coupled with service module placement. At the same time, to further optimize the end-to-end latency of VR video delivery, the routing delay of the rendered VR video stream and the costs of the service module migration are jointly considered with the proposed placement scheme. The goal of this scheme is to minimize the sum of the network costs over a long time under satisfying the delay constraint of each player. We model our strategy as a high-order, nonconvex, and time-varying function. To solve this problem, we transform the placement problem into the min-cut problem by constructing a series of auxiliary graphs. Then, we propose a two-stage iterative algorithm based on convex optimization and graphs theory to solve our object function. Finally, extensive simulation results show that our proposed algorithm can ensure low end-to-end latency for players and low network costs over the other baseline algorithms.

1. Introduction

Wireless virtual reality (VR) games are becoming more and more popular, and it is reported that the global VR gaming market size is projected to reach 45 billion dollars by 2025. A wireless VR game application is generally composed of two parts: a collection module and a service module. The collection module is used to collect the geographic location and actions of players and then delivers the collected information to the service module. The service module encapsulates all the necessary environments to perform logical calculations, render the scene, and synchronize the game information among players [1]. Players of different VR games need different service modules to perform their respective rendering tasks. Players of the same VR game use the same service modules and need to synchronize the information of this VR game with each other (such as character position and score). However, offering low-latency and high-quality VR gaming services to mass wireless players at any time and anywhere is always a major challenge [2–8].

Recently, introducing multiaccess edge computing (MEC) technology to wireless VR games has been a promising computing paradigm to address the above challenges [9–14]. By offloading the rendering tasks from the mobile devices (e.g., VR headsets) to the proximal MEC servers, the players' requirements for ultrahigh computational capacity and strict response latency would be satisfied. Rendering refers to the process of generating images from a model, which is a representation of a 3D object or virtual environment defined by a programming language or data structure. Specifically, since the MEC server has higher computing power than the mobile device, the delay of rendering VR tasks on the MEC server is less than the delay of rendering the same VR tasks on the mobile device [15–19].

However, edge rendering inevitably introduces the edge computing delay and the transmission delay caused by the rendered VR game video stream back to the mobile terminal. Especially, since the data volume of VR video streams is generally huge, the increase in delay will be even more pronounced. Therefore, it is particularly important to optimize the routing of rendered VR game video streams and reasonably allocate the edge resource including wireless spectrum and computation. In addition, it should be noted that deploying service modules on MEC servers increases placement costs, and limited by the storage capacity, service modules of all kinds of VR games cannot deploy on each MEC at the same time [20–23]. But the premise of performing the rendering task of the player on the MEC server is that the service module of the VR game that this user participates in has been deployed on this MEC server [24–26]. Based on the above discussion, the service module placement optimization and the computation resource allocation should be closely coupled to jointly optimize the wireless VR game delivery performance [27–29].

Moreover, in a MEC network scenario of concurrent multiple kinds of wireless VR games, the geographical position of players may change with time, and their access base stations (BSs) may change as they move. To ensure the low routing cost of the rendered VR video streams of one group, the corresponding VR service module serving this group may need to migrate to a new base station. The above situation would increase migration costs [30–34] including hardware wear-and-tear costs and data migration delay costs. Dynamically optimizing the trade-off between the routing cost and migration cost is necessary.

In this paper, we propose a dynamically rendering-aware service module placement scheme. In this scheme, the rendering tasks of VR games are offloaded to the MEC server and closely coupled with service module placement. At the same time, to further optimize the end-to-end latency of VR video delivery, the rendered VR video stream routing delay and service module migration costs are considered with the proposed placement scheme. Specifically, the strategies jointly consider the bandwidth, computing, and storage resource allocation scheme within each time slot and the service module migration cost optimization between different base stations in the adjacent time slot. The goal of this scheme is to minimize the sum of the network costs over a long time under satisfying the delay constraint of each player.

(i) In this paper, we propose a dynamically rendering-aware service module placement scheme, which jointly optimizes service module placement and the associated rendering computation allocation. The goal of this scheme is to minimize the whole network cost based on satisfying the players' low end-to-end delay and high-computing requirements

(ii) We study the problem of how to dynamically place the VR service module to achieve a good balance between the routing delay cost of the rendered VR video stream and the migration cost of the corresponding service module

(iii) We transform our placement problem into the minimal cut problem by developing algebraic conversions and constructing a series of auxiliary graphs. Then, we propose a two-stage iterative algorithm based on convex optimization and graphs theory to solve our objective function within polynomial time

The rest of this paper is organized as follows. Section 2 introduces the system model. Section 3 presents the problem formulation. The proposed solution is presented in Section 4. In Section 5, simulation results are presented and discussed. Finally, the conclusion is given in Section 6.

1.1. Related Work. At present, most of the research on placement strategy focuses on reducing network delay and network overhead for the user by reasonably deploying the services, data, or virtual machines in a suitable location with limited network resources. But the current works ignore considering the dependency relationships between computing and storage. Paper [24] proposes a two-time scale framework that jointly optimizes service placement and request scheduling considering system stability and operation cost. Paper [1] provides a mix of cost models to optimize the deployment of collaborative edge applications to achieve the best overall system performance. Paper [25] proposes a distributed algorithm based on games theory to optimize virtual machine placement in mobile cloud gaming through resource competition to meet the overall requirements of players in a cost-effective manner. Paper [35] proposes a novel offline community discovery and online community adjustment schemes to reduce the internode traffic and the system overhead, which solve the replica placement problem in a scalable and adaptive way. Paper [36] has some similarities with our work, which studies the joint optimization of service placement and request routing in the MEC networks with multidimensional (storage-computation-communication) constraints. In paper [5], the author proposes a MEC-based dynamic cache strategy and an optimized unload strategy to minimize system delay and energy. Paper [27] proposes a rendering-aware tile caching scheme to optimize the end-to-end latency for VR video delivery over multicell MEC networks. Paper [28] designs a view synthesis-based 360 VR caching system to meet the requirements of wireless VR applications and enhance the quality of the VR user experience, which supports MEC and hierarchical caching.

The goal of the recent research on wireless VR mainly focuses on improving the quality of service (QoS), reducing network overhead, or both by proper resource allocation, transcoding technology, introducing edge networks, and etc. Insufficient consideration is given to players' mobility and the network scenario of concurrent multiple kinds of wireless VR games. Paper [4] proposes a blockchain-supported task offloading scheme to resist malicious attacks, which reduces the computing load of virtual machines and

satisfy the high QoE of VR users. Paper [10] proposes a wireless VR network that supports MEC. The network uses a recurrent neural network (RNN) to predict the field of view of each VR user in real-time and transfers the rendering task of VR from the VR device to the MEC server through the rendering model migration function. Paper [16] proposes an adaptive MEC-assisted virtual reality framework, which can adaptively assign real-time virtual reality rendering tasks to MEC servers. Meanwhile, the caching capability of MEC servers can further improve network performance. Paper [37] proposes a task offloading, and resource management scheme based on wireless virtual reality is proposed. The scheme comprehensively considers the factors of cache, computing, and spectrum allocation and minimizes the content delivery delay while guaranteeing quality. Paper [38] studies a multilayer wireless VR video service scenario based on a MEC network. Its main goal is to minimize system energy consumption and delay and to find a balance between these two indicators. Paper [11] proposes to minimize the long-term energy consumption of MEC systems based on THz wireless access by jointly optimizing viewport rendering offloading and downlink transmission power control to support high-quality immersive VR video services. Paper [39] proposes a novel transcoding-enabled VR video caching and delivery framework for edge-enhanced next-generation wireless networks. Paper [40] investigates the optimal wireless streaming of a multi-quality-tiled VR video from a server to multiple users by effectively utilizing characteristics of multi-quality-tiled VR videos and computation resources at the users' side.

2. System Model

The MEC server is a microdata center that is typically deployed with a cellular base station or WiFi access point. Some lightweight virtualization technologies are used to virtualize the hardware resources in the MEC server to realize the flexible sharing of resources.

In this section, as illustrated in Figure 1, we consider a scenario of concurrent multiple kinds of VR games under the cellular network equipped with MEC servers. In this network scenario, there are U players and M base stations (BSs), where each BS is deployed with a MEC server. We represent the set of BSs as $\mathcal{U} = \{1, 2, 3, \cdots, u, \cdots, U\}$ and represent the set of users as $\mathcal{M} = \{1, 2, 3, \cdots, m, \cdots, M\}$. The base stations are connected to each other in a wired way. We assume that there are H kinds of VR games in this scenario, denoted by the set $\mathcal{H} = \{1, 2, 3, \cdots, h, \cdots, H\}$. Therefore, H different service modules are required to support these VR games. In addition, to make dynamic decisions, we model our problem as a time-slotted system, where we use $\mathcal{T} = \{1, 2, 3, \cdots, t, \cdots, T\}$ to denote the set of consecutive time slots under consideration. We assume that each time slot is much larger than the delay caused by transmission and processing.

In the remaining subsections, the mathematical models for communication, dynamic placement, rendering computation, and whole network cost are discussed. Some important notations are summarized in Table 1.

2.1. Placement Cost. In this section, we investigate the dynamic placement scheme of all VR service modules in the system.

We assume that the set of service module placement strategies can be denoted as $\Delta = \{\delta_{mh}^t | m \in \mathcal{M}, h \in \mathcal{H}, t \in \mathcal{T}\}$, where $\delta_{mh}^t = 1$ represents that the VR module service h is storaged in the BS m; otherwise at the time t, $\delta_{mh}^t = 0$.

The cost for using the storage resources when placing service module h on edge node m is characterized by λ_{mh}. The cost of the placement VR service module can be expressed by the following formula:

$$\text{Cost}_P^t = \sum_{m=1}^{M} \sum_{h=1}^{H} \lambda_{mh} \delta_{mh}^t. \tag{1}$$

We assume the storage capacity of BS m is Π_m, and the size of VR service module h is w_h. Due to the total size of the VR service modules deployed in BS m should not exceed the maximum storage capacity of BS m, the constraint should be expressed as

$$\sum_{h=1}^{H} \delta_{mh}^t w_h \le \Pi_m, \forall m \in \mathcal{M}. \tag{2}$$

2.2. Migration Cost. When the players move, due to the changes in the geographical location, the BS that transmits the rendered data to the players may change. At the same time, the BS that originally provided the rendering service for the game group may no longer be the best choice to provide service. The group may need to select a suitable new BS to perform rendering and even may need to deploy the corresponding VR service module on the new selected BS. That is to say, the data information of the service module may need to be migrated from the old MEC server to the new MEC server and built the environment on the new MEC. However, the migration of the VR service module will cause hardware wear-and-tear costs and impose data migration latency costs. The migration delay of each player belonging to the same group is equal and can be expressed as

$$D_{u,t}^{\text{mig}} = \sum_{m=1}^{M} \sum_{h=1}^{H} p_u^h g\left(\delta_{mh}^t, \delta_{mh}^{t-1}\right). \tag{3}$$

In addition, the all migration costs can be expressed as

$$\text{Cost}_M^t = \sum_{m=1}^{M} \sum_{h=1}^{H} \left[f\left(\delta_{mh}^t, \delta_{mh}^{t-1}\right) + g\left(\delta_{mh}^t, \delta_{mh}^{t-1}\right) \right], \tag{4}$$

where $f(\delta_{mh}^t, \delta_{mh}^{t-1})$ and $g(\delta_{mh}^t, \delta_{mh}^{t-1})$ can be, respectively, defined as

$$f\left(\delta_{m,h}^t, \delta_{m,h}^{t-1}\right) = \begin{cases} f_h, & \delta_{mh}^t > \delta_{mh}^{t-1}, \\ 0, & \delta_{mh}^t \le \delta_{mh}^{t-1}, \end{cases} \tag{5}$$

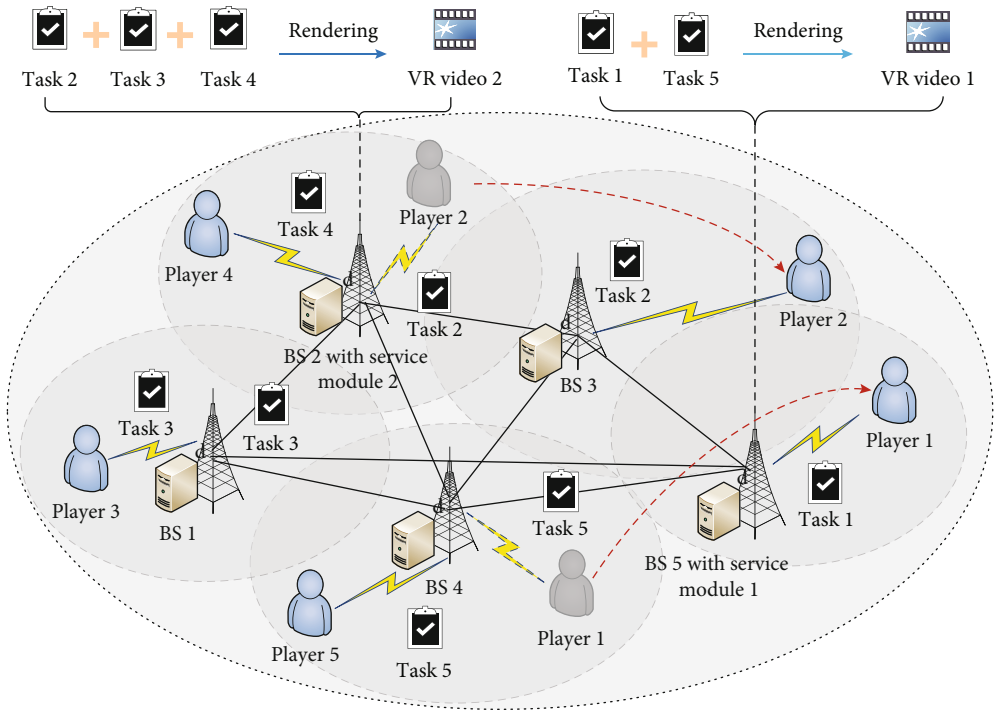


Figure 1: System model. Players 1 and 5 belong to the same VR game and need VR service module 1 to perform rendering. Players 2–4 belong to the same VR game and need VR service module 2 to perform rendering. Among them, the player 1 migrates from the coverage of BS 4 to the coverage of BS 5, and the player 2 migrates from the coverage of BS 2 to the coverage of BS 3. Player 3 access to BS 1 and offload the task 3 from BS 1 to BS 2.

Table 1: List of key notations.

Notation	Definition
$\mathcal{M}$	Set of BSs
$\mathcal{U}$	Set of users
$\mathcal{H}$	Set of VR games
$\mathcal{T}$	Set of consecutive time slots
δ_{mh}^t	Placement indicator
Π_m	The maximum storage capability of BS m
σ^2	The variance of additive white Gaussian noise
B^t	The maximum bandwidth of BS at time t
K_m	The maximum computing capability of BS m
a_{mu}^t	The access indicator at the time t
s_{mh}^t	The BS selection indicator at the time t
$d_{m,m'}$	The delay of routing one bit of data from BS m to BS m'
p_u^h	The indicator of player u whether joining in group$h$

$$g\left(\delta_{m,h}^t, \delta_{m,h}^{t-1}\right) = \begin{cases} \upsilon g_h, & \delta_{mh}^t > \delta_{mh}^{t-1}, \\ 0, & \delta_{mh}^t \le \delta_{mh}^{t-1}, \end{cases} \tag{6}$$

where g_h represents the migration delay of the VR service module h and f_h represents the cost of reconfiguring the VR service module h. To be reasonable, we make the values of $f(\delta_{mh}^t, \delta_{mh}^{t-1})$ and $g(\delta_{mh}^t, \delta_{mh}^{t-1})$ to the same order of magnitude by adjusting the parameter υ.

2.3. Rendering Cost. Players in the same group may have overlapping computational tasks; in this section, we assume that the MEC server computes centrally after collecting all the information of the players in the group. Therefore, we allocate the computing resources on each server by the group.

In the MEC network, when the MEC server is serving only one group, that group can certainly get more computing resources to perform rendering, resulting in a low processing latency experience. However, in general, each MEC server needs to serve multiple groups at the same time, which can lead to competition for computation resources. In particular, if too many groups render on the same MEC server, the delays for all groups connected to this server will increase dramatically.

$p_u^h \in \{0, 1\}$ is the indicator, to represents whether the players u join in the VR game h. Due to one player can only join in one kind of game, so the corresponding constraints can be, respectively, formulated as

$$\sum_{h=1}^{H} p_u^h = 1, \forall u \in \mathcal{U}. \tag{7}$$

We use $\mathcal{S} = \{s_{mh}^t | m \in \mathcal{M}, h \in \mathcal{H}, t \in \mathcal{T}\}$ to denote the set of the rendering base station selection strategies. When the

group h selects the MEC server m to perform the rendering task, $s_{mh}^t = 1$ at the time t; otherwise, $s_{mh}^t = 0$.

In order to ensure the information synchronization between users in the same group, we assume that a group can only select one MEC server to process tasks at a time slot, so the corresponding constraints can be formulated as

$$\sum_{m=1}^{M} s_{mh}^t = 1, \forall h \in \mathcal{H}, t \in \mathcal{T}. \tag{8}$$

Since the cost of putting the VR service module on the server is high, we put the VR service module on the BS, which has been selected to process the groups' tasks. So, we can get the following formula:

$$\delta_{mh}^t = s_{mh}^t, \forall t \in \mathcal{T}, h \in \mathcal{H}, m \in \mathcal{M}. \tag{9}$$

We assume that the maximum computing capability of the MEC server m is K_m (Hz) and the computing resource of the BS m allocated to group h at time t is k_{mh}^t. We use $\mathcal{K} = \{k_{mh}^t | m \in \mathcal{M}, h \in \mathcal{H}, t \in \mathcal{T}\}$ to represent the computing resource allocation scheme. C_h^t represents the computing resource needed for group h at time t. The rendering delay of players belonging to the same group is equal. So, the rendering delay of player u at time slot t can be expressed as

$$D_{u,t}^{\text{rend}} = \sum_{m=1}^{M} \sum_{h=1}^{H} s_{mh}^t p_u^h \frac{C_h^t}{k_{mh}^t}, t \in \mathcal{T}. \tag{10}$$

So, the rendering cost can be denoted by the sum of the rendering latency of all groups, which can be expressed by

$$\text{Cost}_R^t = \sum_{m=1}^{M} \sum_{h=1}^{H} s_{mh}^t \frac{C_h^t}{k_{mh}^t}, t \in \mathcal{T}. \tag{11}$$

At the same time, a MEC server cannot allocate more computing resources to the groups; it serves than its maximum computing resources. Therefore, the corresponding computing resources constraints can be formulated as

$$\sum_{h=1}^{H} k_{mh}^t \le K_m, \forall m \in \mathcal{M}, t \in \mathcal{T}. \tag{12}$$

2.4. Communication Cost. In this section, we present the communication model in the mobile edge computing networks based on mmWave, which concentrates on the downlink transmission. At the same time, we introduce the routing transmission delay.

2.4.1. Downlink Delay. We use $\mathcal{A} = \{a_{mu}^t | m \in \mathcal{M}, u \in \mathcal{U}, t \in \mathcal{T}\}$ as the access scheme, where the $a_{mu}^t = 1$ means that players u is associated with BS m at the time t to obtain the rendered game video stream, while $a_{mu}^t = 0$ denotes that players u is not served by BS m at the time t.

Moreover, players cannot connect to multiple base stations at the same time, and we need to ensure that each player can connect to a suitable one. So we get the following constraint formula:

$$\sum_{m=1}^{M} a_{mu}^t = 1, \forall u \in \mathcal{U}. \tag{13}$$

We adopt the orthogonal spectrum reuse scheme in this system; i.e., all BS share the total frequency bandwidth, and there is no interference between the users served by the same BS. The data amount of the uplink transmission is small, only including some players' information, such as commands and actions. So, the delay and cost of this process are ignored in this paper.

The downlink transmission is used to transmit the rendered VR video stream, in which the amount of data is larger. Therefore, millimeter Wave technology with large bandwidth is adopted for downlink transmission. Assume that all channels are subject to independent identically distributed quasistatic Rayleigh block fading. The path loss can be expressed as follow:

$$L_{mu}^t = \eta^t \left(\left| d_{mu}^t \right|^{-\varsigma^t} \right), \tag{14}$$

where η^t is the downlink constant related to frequency, ς^t is the downlink path loss exponent at time t, and $|d_{mu}^t|$ is the distance between the players u and BS m at time t.

Millimeter wave has the characteristics of short wavelength, small power, and directional antenna. The interference between the same frequency beam can be reduced well by millimeter wave interference cancelation technology. As the interference cancelation technology is not the focus of this paper and the millimeter transmission tends to be noise-limited and weak-interference, the interference in the transmission process of millimeter waves is ignored in this paper by referring to papers [16, 41, 42]. So, the signal-to-interference-plus-noise ratio received by the players u from the BS u is expressed as follows:

$$\text{SINR}_{mu}^t = \frac{p_{mu} g_{mu}^t L_{mu}^t}{\sigma^2}, \tag{15}$$

where g_{mu}^t is the downlink antenna gain using direction beamforming between players u and BS m at the time t, p_{mu} is the transmission power between players u and BS m, and σ^2 is the variance of additive white Gaussian noise (AWGN).

We assume that the spectrum bandwidth allocated to players u from BS m at time t is B_{mu}^t and use $\mathcal{B} = \{B_{mu}^t | m \in \mathcal{M}, u \in \mathcal{U}, t \in \mathcal{T}\}$ as the bandwidth allocation scheme. Since the total bandwidths that the BS m allocates to its access players do not exceed the whole bandwidths in the wireless access network at time t, which is B^t, corresponding bandwidth constraints can be formulated as

$$\sum_{u=1}^{U} B_{mu}^t \le B^t, \forall m \in \mathcal{M}, t \in \mathcal{T}. \tag{16}$$

Then, the uplink transmission rate between the players u and the BS m at time t is

$$r_{mu}^{t} = B_{mu}^{t}\log_2\left(1 + \mathrm{SINR}_{mu}^{t}\right). \tag{17}$$

We assume that the size of the video images needed to transmit to the players u at time t is o_u^t, so the delay of downlink transmission for players u at time t is

$$D_{u,t}^{\mathrm{down}} = \sum_{m=1}^{M} a_{mu}^{t}\frac{o_u^t}{r_{mu}^{t}}. \tag{18}$$

The delay of downlink transmission for all players at time t, i.e., the downlink communication cost of the network, is

$$E_1^t = \sum_{u=1}^{U} D_{u,t}^{\mathrm{down}}. \tag{19}$$

2.4.2. Routing Delay. In this section, we divided the players into H groups based on the differences in VR games they participate in. Different groups need different service modules to perform rendering. We need to select an appropriate MEC server to perform rendering for group h and route the rendered video stream quickly to the access base station of the user belonging to the group h. The selected MEC server needs to have deployed the corresponding VR service modules and has sufficient computing resources to perform rendering tasks.

According to the above assumption, at the time slot t, the delay of routing the rendered VR content requested by user u from the working (rendering) BS m to this user's access BS m' can be expressed as

$$D_{u,t}^{\mathrm{rout}} = \sum_{m=1}^{M}\sum_{h=1}^{H}\sum_{m'=1}^{M} p_u^h a_{m'u}^{t} s_{mh}^{t} d\left(m, m'\right) o_u^t, \tag{20}$$

where $d(m, m')$ is the delay of routing one bit of data from BS m to BS m', when $m = m'$, $d(m, m') = 0$.

The routing delay of all players at time t, i.e., the routing cost of the network, is

$$E_2^t = \sum_{u=1}^{U} D_{u,t}^{\mathrm{rout}}. \tag{21}$$

So, the communication cost at time t can be expressed as the sum of downlink transmission delay and routing delay.

$$\mathrm{Cost}_C^t = E_1^t + E_2^t, t \in \mathcal{T}. \tag{22}$$

3. Problem Formulation

Our goal is to develop dynamical service module placement strategies based on rendering-aware. The goal of those strategies is to minimize the sum of the whole network costs over a long time under satisfying the delay constraint of each player. The strategies jointly consider the resource allocation scheme within each time slot and the service module migration scheme between different base stations in the adjacent time slot.

We assume that the maximum tolerance delay of the group u is $\mathscr{D}_u$. According to the above formula, the actual end-to-end delay of player u at time slot t can be expressed by the following:

$$\mathscr{D}_{u,t}{}' = D_{u,t}^{\mathrm{down}} + D_{u,t}^{\mathrm{rout}} + D_{u,t}^{\mathrm{rend}} + D_{u,t}^{\mathrm{mig}}. \tag{23}$$

We define $\varepsilon_1 - \varepsilon_4$ as the weight coefficients, which represent the proportion of communication cost, rendering cost, placement cost, and migration cost in the objective function, respectively. So, the optimization problem can be formulated as follows:

$$\begin{aligned}
\Gamma_1 : \quad & \min_{\mathscr{A},\mathscr{S},\mathscr{B},\mathscr{K},\Delta} \sum_{t=1}^{T} \varepsilon_1 \mathrm{Cost}_C^t + \varepsilon_2 \mathrm{Cost}_R^t + \varepsilon_3 \mathrm{Cost}_P^t + \varepsilon_4 \mathrm{Cost}_M^t \\
\text{s.t.} \quad & C1 : \sum_{m=1}^{M} a_{mu}^{t} = 1, \forall u \in \mathscr{U}, t \in \mathcal{T} \\
& C2 : \sum_{m=1}^{M} s_{mh}^{t} = 1, \forall h \in \mathscr{H}, t \in \mathcal{T} \\
& C3 : \sum_{u=1}^{U} B_{mu}^{t} \le B^{t}, \forall m \in \mathscr{M}, t \in \mathcal{T} \\
& C4 : \sum_{h=1}^{H} k_{mh}^{t} \le K_m, \forall m \in \mathscr{M}, t \in \mathcal{T} \\
& C5 : \sum_{h=1}^{H} \delta_{mh}^{t} w_h \le \Pi_m, \forall m \in \mathscr{M} \\
& C6 : \mathscr{D}'_{u,t} \le \mathscr{D}_u \\
& a_{mu}^{t}, s_{mh}^{t}, \delta_{mh}^{t} \in \{0, 1\}.
\end{aligned} \tag{24}$$

Constraint C_1 ensures that a player cannot connect to multiple base stations at the same time; meanwhile, each user can connect to a BS. Constraint C_2 ensures that a group can only select one MEC server to perform rendering tasks at a time slot. Constraint C_3 ensures that the total bandwidths that the BS m allocates to its access players do not exceed the whole bandwidths in the wireless access network at time t. Constraint C_4 ensures that a MEC server cannot allocate more computing resources to the groups; it serves than its maximum computing resources. Constraint C_5 ensures that the total size of the VR service modules storage in BS m should not exceed the maximum storage capacity of BS m. Constraint C_6 ensures the total delay of each group cannot exceed its maximum tolerance delay.

4. Solution

In this section, in order to solve the original problem efficiently, we decompose the original problem into two subproblems including dynamic access and service module placement scheme and the quasistatic resource allocation. Then, we use minimum cut theory and convex optimization to solve the above subproblems, respectively.

4.1. Problem Reformulation. Firstly, to get rid of constraint 1 and constraint 2, we redefine sets $\mathscr{A} = \{a_{mu}^t | m \in \mathscr{M}, u \in \mathscr{U}, t \in \mathscr{T}\}$ and $\mathscr{S} = \{s_{mh}^t | m \in \mathscr{M}, h \in \mathscr{H}, t \in \mathscr{T}\}$ as $\mathscr{A}_* = \{a_{u*}^t, u \in \mathscr{U}, t \in \mathscr{T}\}$ and $\mathscr{S}_* = \{s_{h*}^t, h \in \mathscr{H}, t \in \mathscr{T}\}$, respectively, where $\mathscr{A}_*^t = \{a_{u*}^t, u \in \mathscr{U}\}$ is the set of access decisions at time t and $a_{u*}^t \in \mathscr{M}$ represents the BS accessed by the players u, and there is a one-to-one mapping relationship between it and the set $\mathscr{A}_u^t = \{a_{mu}^t, m \in \mathscr{M}\}$. That is, $a_{mu}^t = 1$ and $\{a_{iu}^t = 0 | i \in \mathscr{M}, i \neq m\}$ when $a_{u*}^t = m$. This way of coding can satisfy the constraint $C1$ that a player can only access one base station at the same time.

In the same way, $\mathscr{S}_*^t = \{s_{h*}^t, h \in \mathscr{H}\}$ is the set of BS selection scheme at time t. $s_{h*}^t \in \mathscr{M}$ represents BS serving group h at time t, and there is a one-to-one mapping relationship between it and the set $\mathscr{S}_h^t = \{s_{mh}^t, m \in \mathscr{M}\}$. This way of coding can satisfy the constraint $C2$ that a group can only select one MEC server to perform editing tasks at a time slot.

So, the δ_{mh}^t can be redefined as

$$\delta_{mh}^t = \begin{cases} 1, & s_{h*}^t = m, \forall h \in \mathscr{H}, \\ 0, & \text{otherwise.} \end{cases} \tag{25}$$

Moreover, $\mathscr{B}_*^t = \{B_{u*}^t, u \in \mathscr{U}\}$ is the set of bandwidth allocation scheme at time t. $B_{u*}^t = B_{a_{u*}^t u}^t \in [0, B^t]$ is the bandwidth that BS a_{u*}^t allocate to the players u at time t. $\mathscr{K}_*^t = \{k_{h*}^t, h \in \mathscr{H}\}$ is the set of computing resources allocation scheme at time t. $k_{h*}^t = k_{s_{h*}^t h}^t \in [0, K_m]$ is the computing resources that BS s_{h*}^t allocate to the group h at time t.

Thus, we transform the original problem into the following problem:

$$\begin{aligned} \Gamma_2 : \min_{\mathscr{A}_*, \mathscr{S}_*, \mathscr{B}_*, \mathscr{K}_*} & \sum_{t=1}^{T} \left(\varepsilon_1 \sum_{u=1}^{U} \frac{o_u^t}{B_{a_{u*}^t u}^t log_2\left(1 + SINR_{u*}^t\right)} \right. \\ & + \varepsilon_1 \sum_{h=1}^{H} \sum_{u=1}^{U} p_u^h d\left(s_{h*}^t, a_{u*}^t\right) + \varepsilon_2 \sum_{h=1}^{H} \frac{C_h^t}{k_{s_{h*}^t h}^t} \\ & \left. + \varepsilon_3 \sum_{m=1}^{M} \sum_{h=1}^{H} \lambda_{mh} 1\left(s_{h*}^t = m\right) + \varepsilon_4 \sum_{h=1}^{H} \left[f\left(s_{h*}^t, s_{h*}^{t-1}\right) + g\left(s_{h*}^t, s_{h*}^{t-1}\right) \right] \right) \\ C4' : & \sum_{h \in H_m^t} k_{h*}^t \leq K_m, \forall m \in \mathscr{M}, t \in \mathscr{T} \\ C5' : & \sum_{h=1}^{H} \delta_{mh}^t w_h \leq \Pi_m, \forall m \in \mathscr{M} \\ C6' : & \mathscr{D}'_{ut} \leq \mathscr{D}_u \\ & a_{u*}^t, s_{h*}^t, \in \mathscr{M}, \end{aligned} \tag{26}$$

where H_m^t represents the set of all the groups that render on the BS m and U_m^t represents the set of all the players that access the BS m at time t. Constraint 5 can be satisfied by the k-size minimum cut algorithm. $1(.)$ is a binary function that equals 1 if the specified condition holds and 0 otherwise, where A is the penalty function, which can be expressed as $\mathscr{D}'_{ut}$:

$$\begin{aligned} \mathscr{D}'_{ut} = & \sum_{h=1}^{H} p_u^h \left[g\left(s_{h*}^t, s_{h*}^{t-1}\right)\right) + \frac{C_h^t}{k_{s_{h*}^t h}^t} + d\left(s_{h*}^t, a_{u*}^t\right) \Bigg] \\ & + \frac{o_u^t}{B_{a_{u*}^t u}^t log_2\left(1 + SINR_{u*}^t\right)}. \end{aligned} \tag{27}$$

Due to our objective function containing dynamic optimization and quasistatic optimization, we divide the target function into two parts.

For the part one,

$$\begin{aligned} \text{Cost}_I = \sum_{t=1}^{T} \Bigg(& \varepsilon_1 \sum_{h=1}^{H} \sum_{u=1}^{U} p_u^h d\left(s_{h*}^t, a_{u*}^t\right) + \varepsilon_3 \sum_{m=1}^{M} \sum_{h=1}^{H} \lambda_{mh} 1\left(s_{h*}^t = m\right) \\ & + \varepsilon_4 \sum_{h=1}^{H} \left[f\left(s_{h*}^t, s_{h*}^{t-1}\right) + g\left(s_{h*}^t, s_{h*}^{t-1}\right) \right] \Bigg). \end{aligned} \tag{28}$$

We design an iterative algorithm to update the access decisions of players and the placement schemes of the VR service module in each round by performing an operation called α expansion. Furthermore, we optimize the expansion by minimizing graph cuts.

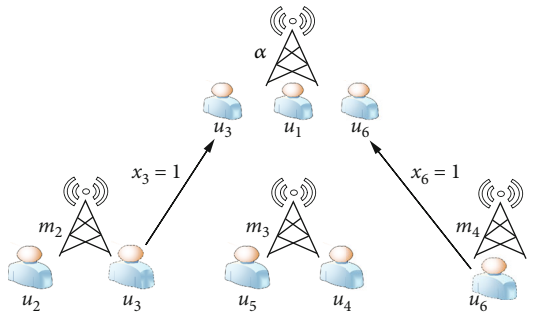
For the part two,

$$\text{Cost}_{II} = \sum_{t=1}^{T} \left(\varepsilon_1 \sum_{u=1}^{U} \frac{o_u^t}{B_{u*}^t \log_2\left(1 + \text{SINR}_{u*}^t\right)} + \varepsilon_2 \sum_{h=1}^{H} \frac{C_h^t}{k_{h*}^t} \right). \tag{29}$$

We use convex optimization to solve the resource allocation problem at each time slot.

4.2. Optimizing Dynamic Access and Placement Strategies by Graph Cuts. In this section, we introduce the α expansion algorithm and how to construct a helper graph and encode the costs of part I into weights on the graph edges. Then, we demonstrate that the min-cut of the graph corresponds to the optimal decisions for the α expansion.

4.2.1. α Expansion. An α expansion can be defined as a binary optimization and reflects the trend of moving the module served for group h from the current base station to the base station α and the trend of users accessing base station α from the current base station. As shown in Figure 2, when we selected BS α as the expansion, $a_{u*}^{t\alpha}$ has a binary choice to stay as $a_{u*}^{t\alpha} = a_{u*}^t$ or change to $a_{u*}^{t\alpha} = \alpha$. In the same way, $s_{h*}^{t\alpha}$ has a binary choice to stay as $s_{h*}^{t\alpha} = s_{h*}^t$ or change to $s_{h*}^{t\alpha} = \alpha$.



	u_1	u_2	u_3	u_4	u_5	u_6
Before	α	m_2	m_2	m_3	m_3	m_4
After	α	m_2	α	m_3	m_3	α
x	1	0	1	0	0	1

Figure 2: α expansion. The player u_3 changes the access base station from m_2 to α, and u_6 changes the access base station from m_4 to α, respectively.

For the sake of calculation, the resultant after expansion also can be expressed by two indicator vectors with binary decision variables. (1) $x't = \{x_1't, \cdots, x_u't\}$, where for all $u \in U$, we define $x_u't = 1$ if $a_{u*}^{t\alpha} = \alpha$; otherwise, $x_u't = 0$. (2) $x = \{x_1^t, \cdots, x_h^t\}$, where for all $h \in H$, we define $x_h^t = 1$ if $s_{h*}^{t\alpha} = \alpha$; otherwise, $x_h = 0$. Note that, if the module served for group h is already on BS α, $x_h^t = 1$, if the players u is already access BS α, $x_u't = 1$.

4.2.2. Transforming the Cost$_I$. After performing an "α expansion," we reconstruct the Cost$_I$ as Cost$_I^\alpha$ using binary variables $x_u^{t'}$ and x_h^t; at the same time, we define $\bar{x}_u't = 1 - x_u't$ and $\bar{x}_h^t = 1 - x_h^t$. And we can get

$$\begin{aligned}
&\varepsilon_1 \sum_{t=1}^{T}\sum_{h=1}^{H}\sum_{u=1}^{U} p_u^h o_u^t d\left(s_{h*}^t, a_{u*}^t\right)^\alpha \\
&\quad = \varepsilon_1 \sum_{t=1}^{T}\sum_{h=1}^{H}\sum_{u=1}^{U} p_u^h o_u^t \Big[d\left(s_{h*}^t, a_{u*}^t\right)\bar{x}_h^t \bar{x}_u' t \\
&\qquad + d\left(\alpha, a_{u*}^t\right) x_h^t \bar{x}_u' t + d\left(s_{h*}^t, \alpha\right)\bar{x}_h^t x_u^{t'} \Big],
\end{aligned} \tag{30}$$

$$\begin{aligned}
\varepsilon_4 \sum_{t=1}^{T}\sum_{h=1}^{H} f\left(s_{h*}^t, s_{h*}^{t-1}\right)^\alpha &= \varepsilon_4 \sum_{t=1}^{T}\sum_{h=1}^{H} \Big[f\left(s_{h*}^t, s_{h*}^{t-1}\right)\bar{x}_h^t \bar{x}_h^{t-1} \\
&\quad + f\left(\alpha, s_{h*}^{t-1}\right) x_h^t \bar{x}_h^{t-1} + f\left(s_{h*}^t, \alpha\right)\bar{x}_h^t x_h^{t-1} \Big],
\end{aligned} \tag{31}$$

$$\begin{aligned}
\varepsilon_4 \sum_{t=1}^{T}\sum_{h=1}^{H} g\left(s_{h*}^t, s_{h*}^{t-1}\right)^\alpha &= \varepsilon_4 \sum_{t=1}^{T}\sum_{h=1}^{H} \Big[g\left(s_{h*}^t, s_{h*}^{t-1}\right)\bar{x}_h^t \bar{x}_h^{t-1} \\
&\quad + g\left(\alpha, s_{h*}^{t-1}\right) x_h^t \bar{x}_h^{t-1} + g\left(s_{h*}^t, \alpha\right)\bar{x}_h^t x_h^{t-1} \Big].
\end{aligned} \tag{32}$$

Then, based on the definition of δ_{mh}^t, we can rewrite it as

$$\varepsilon_3 \sum_{m=1}^{M}\sum_{h=1}^{H} \lambda_{mh} 1\left(s_{h*}^t = m\right)^\alpha = \sum_{m=1}^{M}\sum_{h=1}^{H} \left[\lambda_{\alpha h} x_h^t + \lambda_{mh}\bar{x}_h^t\right]. \tag{33}$$

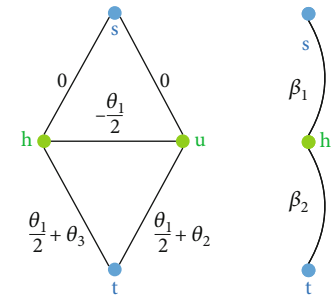


Figure 3: Graph construction. The first figure correspond to $\theta_1\bar{x}_u\bar{s}_h + \theta_2\bar{x}_u + \theta_3\bar{s}_h$; the last figure corresponds to $\beta_1 s_h + \beta_2\bar{s}_h$.

4.2.3. A Simple Example of Graph Cut. Based on the derivation above, we find that $\sum_{t=1}^{T}\sum_{h=1}^{H}\sum_{u=1}^{U} p_u^h o_u^t d(s_{h*}^t, a_{u*}^t)^\alpha$ and $\sum_{t=1}^{T}\sum_{h=1}^{H}\sum_{u=1}^{U} f(s_{h*}^t, s_{u*}^{t-1})^\alpha$ correspond to the sum of the products of pairs of binary variables; $\sum_{m=1}^{M}\sum_{h=1}^{H}\lambda_{mh}1(s_{h*}^t = m)^\alpha$ corresponds to the sum of binary variables.

Taking $\theta_1\bar{x}_u\bar{s}_h + \theta_2\bar{x}_u + \theta_3\bar{s}_h$ and $\beta_1 s_h + \beta_2\bar{s}_h$ as simple examples, we next will introduce how to minimize them, respectively, by constructing a graph. The basic idea is to construct a helper graph to make the sum of the weights of the min-cut of the graph equal the optimal value of the objective function. The above cut edges divide the nodes in the graph into two parts: one part of the nodes is on the side of node s, and the corresponding value is 0. The other part of the nodes is on the side of node t, and the corresponding value is 1. In addition, the minimum cut can be computed in polynomial time only if all the edge weights are nonnegative. Next, we will introduce how to build a diagram for our example.

For $\theta_1\bar{x}_u\bar{s}_h + \theta_2\bar{x}_u + \theta_3\bar{s}_h$, we reformulate the expression to construct each edge in a subgraph.

$$\begin{aligned}
&\theta_1\bar{x}_u\bar{s}_h + \theta_2\bar{x}_u + \theta_3\bar{s}_h \\
&\quad = \frac{\theta_1}{2}\bar{x}_u\bar{s}_h + \frac{\theta_1}{2}\bar{x}_u\bar{s}_h + \theta_2\bar{x}_u + \theta_3\bar{s}_h \\
&\quad = -\frac{\theta_1}{2}x_u\bar{s}_h - \frac{\theta_1}{2}\bar{x}_u s_h + \left(\frac{\theta_1}{2} + \theta_2\right)\bar{x}_u + \left(\frac{\theta_1}{2} + \theta_3\right)\bar{s}_h.
\end{aligned} \tag{34}$$

As illustrated in the first figure in Figure 3, the weight of edge between node u and node h is $-\theta_1/2$, the weight of edge between node u and node t is $\theta_1/2 + \theta_2$, and the weight of edge between node h and node t is $\theta_1/2 + \theta_3$, where $-\theta_1/2 \geq 0$. For example, when we divide the first graph's nodes in Figure 3 into two parts by cutting the edge between nodes s and h, the edge between nodes h and u, and the edge between nodes u and t, node u and node s are in the same part, and node h and node t are in the same part (i.e., $x_u = 0$, $s_h = 1$, and $\bar{x}_u = 1$, $\bar{s}_h = 0$). The value of the first graph function is $\theta_1\bar{x}_u\bar{s}_h + \theta_2\bar{x}_u + \theta_3\bar{s}_h = \theta_2$, which is equal to the sum of the weights of the cut edges. In the last figure in Figure 3, the weight of edge between node h and node s is β_1, and the weight of edge between node h and node t is β_2.

4.2.4. Constructing a Graph to Solve the Subproblem. In this section, we construct a graph $\mathcal{G} \ll (\mathcal{V}, \mathcal{E})$ to make the sum of the edges' weights in the minimal cut set equals the optimal value of our objective function. In this graph, there are $T * U$ vertices corresponding to the players, and $T * H$ vertices corresponding to the groups. Moreover, a source vertex s and a terminal vertex t are also in the vertex set. As a result, the set of vertices in $\mathcal{G}$ is given by $\{x_u' t | u \in \mathcal{U}, t \in \mathcal{T}\} \cup \{x_h^t | h \in \mathcal{H}, t \in \mathcal{T}\} \cup \{s, t\}$.

In the next section, we add edges to the graph and give each edge an appropriate weight. Firstly, based on the example of the last figure in Figure 3. The weights of the edges between node x_h^t and node s can be represented as $\lambda_{\alpha h}$, and the weights of the edges between node x_h^t and node t can be represented as λ_{mh}.

Next, we rewrite formulas (30) and (31) to formulas (40) and (41) based on the example of the first figure in Figure 3.

Therefore, the weight of the edge between the vertex $x_u' t$ and vertex x_h^t is

$$p_u^h o_u^t \frac{d(\alpha, a_{u*}^t) + d(s_{h*}^t, \alpha) - d(s_{h*}^t, a_{u*}^t)}{2}, \tag{35}$$

where $d(\alpha, a_{u*}^t) + d(s_{h*}^t, \alpha) - d(s_{h*}^t, a_{u*}^t)$ is always satisfied, which can be proved by the triangle inequality.

In the same way, the weight of the edge between the vertex x_h^{t-1} and vertex x_h^t is

$$\frac{f(\alpha, s_{h*}^{t-1}) + f(s_{h*}^t, \alpha) - f(s_{h*}^t, s_{h*}^{t-1})}{2}, \tag{36}$$

where $f(\alpha, s_{h*}^{t-1}) + f(s_{h*}^t, \alpha) - f(s_{h*}^t, s_{h*}^{t-1})$ is always satisfied, which can be proved by the triangle inequality.

In addition, based on the above derivation, we can also get that the partial of weight of the edge between vertex x_h^t and vertex t is

$$\frac{d(s_{h*}^t, a_{u*}^t) - d(\alpha, a_{u*}^t) + d(s_{h*}^t, \alpha)}{2}. \tag{37}$$

The partial of weight of the edge between vertex $x_u' t$ and vertex t is

$$\frac{d(s_{h*}^t, a_{u*}^t) + d(\alpha, a_{u*}^t) - d(s_{h*}^t, \alpha)}{2}. \tag{38}$$

Moreover, the partial of weight of the edge between vertex x_h^t and vertex t is

$$\frac{-f(\alpha, s_{h*}^{t-1}) + f(s_{h*}^t, \alpha) + f(s_{h*}^t, s_{h*}^{t-1})}{2} + \frac{f(\alpha, s_{h*}^{t-1}) - f(s_{h*}^t, \alpha) + f(s_{h*}^t, s_{h*}^{t-1})}{2}. \tag{39}$$

Therefore, we can perform the following transformation of the objective function based on the above analysis:

$$\begin{aligned}
&\varepsilon_1 \sum_{t=1}^{T} \sum_{h=1}^{H} \sum_{u=1}^{U} p_u^h o_u^t d(s_{h*}^t, a_{u*}^t)^{\alpha} \\
&= \varepsilon_1 \sum_{t=1}^{T} \sum_{h=1}^{H} \sum_{u=1}^{U} p_u^h o_u^t \Big[d(s_{h*}^t, a_{u*}^t) \bar{x}_h^t \bar{x}_u' t \\
&\quad + d(\alpha, a_{u*}^t)(1 - \bar{x}_h^t) \bar{x}_u' t + d(s_{h*}^t, \alpha) \bar{x}_h^t \left(1 - \bar{x}_u^{t'}\right) \Big] \\
&= \varepsilon_1 \sum_{t=1}^{T} \sum_{h=1}^{H} \sum_{u=1}^{U} p_u^h o_u^t \Big[d(\alpha, a_{u*}^t) \bar{x}_u' t + d(s_{h*}^t, \alpha) \bar{x}_h^t \\
&\quad + \left(d(s_{h*}^t, a_{u*}^t) - d(\alpha, a_{u*}^t) - d(s_{h*}^t, \alpha) \right) \bar{x}_h^t \bar{x}_u' t \Big] \\
&= \varepsilon_1 \sum_{t=1}^{T} \sum_{h=1}^{H} \sum_{u=1}^{U} p_u^h o_u^t \Bigg[\frac{d(\alpha, a_{u*}^t) + d(s_{h*}^t, \alpha) - (d(s_{h*}^t, a_{u*}^t)}{2} \bar{x}_h^t x_u' t \\
&\quad + \frac{d(\alpha, a_{u*}^t) + d(s_{h*}^t, \alpha) - d(s_{h*}^t, a_{u*}^t)}{2} x_h^t \bar{x}_u' t \\
&\quad + \frac{d(s_{h*}^t, a_{u*}^t) + d(\alpha, a_{u*}^t) - d(s_{h*}^t, \alpha)}{2} \bar{x}_u' t \\
&\quad + \frac{d(s_{h*}^t, a_{u*}^t) - d(\alpha, a_{u*}^t) + d(s_{h*}^t, \alpha)}{2} \bar{x}_h^t \Bigg],
\end{aligned} \tag{40}$$

$$\begin{aligned}
&\varepsilon_4 \sum_{t=1}^{T} \sum_{h=1}^{H} f(s_{h*}^t, s_{h*}^{t-1})^{\alpha} \\
&= \varepsilon_4 \sum_{t=1}^{T} \sum_{h=1}^{H} \Big[f(s_{h*}^t, s_{h*}^{t-1}) \bar{x}_h^t \bar{x}_h^{t-1} \\
&\quad + f(\alpha, s_{h*}^{t-1})(1 - \bar{x}_h^t) \bar{x}_h^{t-1} + f(s_{h*}^t, \alpha) \bar{x}_h^t (1 - \bar{x}_h^{t-1}) \Big] \\
&= \varepsilon_4 \sum_{t=1}^{T} \sum_{h=1}^{H} \Big[f(\alpha, s_{h*}^{t-1}) \bar{x}_h^{t-1} + f(s_{h*}^t, \alpha) \bar{x}_h^t + \left(f(s_{h*}^t, s_{h*}^{t-1}) \right) \\
&\quad - f(\alpha, s_{h*}^{t-1}) - f(s_{h*}^t, \alpha)) \bar{x}_h^t \bar{x}_h^{t-1} \Big] \\
&= \varepsilon_4 \sum_{t=1}^{T} \sum_{h=1}^{H} \Bigg[\frac{f(\alpha, s_{h*}^{t-1}) + f(s_{h*}^t, \alpha) - f(s_{h*}^t, s_{h*}^{t-1})}{2} x_h^t \bar{x}_h^{t-1} \\
&\quad + \frac{f(\alpha, s_{h*}^{t-1}) + f(s_{h*}^t, \alpha) - f(s_{h*}^t, s_{h*}^{t-1})}{2} \bar{x}_h^t x_h^{t-1} \\
&\quad + \frac{-f(\alpha, s_{h*}^{t-1}) + f(s_{h*}^t, \alpha) + f(s_{h*}^t, s_{h*}^{t-1})}{2} \bar{x}_h^t \\
&\quad + \frac{f(\alpha, s_{h*}^{t-1}) - f(s_{h*}^t, \alpha) + f(s_{h*}^t, s_{h*}^{t-1})}{2} \bar{x}_h^{t-1} \Bigg].
\end{aligned} \tag{41}$$

The detail process of the auxiliary diagram construction is concluded in Algorithm 1.

4.3. Resource Allocation Scheme Based on Convex Optimization. In this section, we mainly focus on the optimization of Cost_{II}, that is, minimizing the total transmission and editing delay in each time interval through the reasonable allocation of computing and spectrum resources. When $\mathcal{A} *$ and $\mathcal{S} *$ are determined, the original optimization problem can be expressed in the following form:

Input: The network delay between BS m and BS m' $d(m, m')$; The switching cost of group h at time t $f(s_{h*}^t, s_{h*}^{t-1})$; The migration delay of group h at time t $g(s_{h*}^t, s_{h*}^{t-1})$;
Output: The value of binary variables $x_u't$ and x_h^t; The auxiliary graph $\mathcal{G} = (\mathcal{E}, \mathcal{V})$; the variables $s_{h*}^{t\alpha}$ and $a_{u*}^{t\alpha}$

Initialization $\mathcal{V} = \{x_u't | u \in \mathcal{U}, t \in \mathcal{T}\} \cup \{x_h^t | h \in \mathcal{H}, t \in \mathcal{T}\} \cup \{source, terminal\}$; $\mathcal{E} = 0$;
for $t = 1 : T$ **do**
  for $h = 1 : H$ **do**
    $u = 1 : U$ **do**
      $e(x_h^t, x_u't) = p_u^h o_u^t (d(\alpha, a_{u*}^t) + d(s_{h*}^t, \alpha) - d(s_{h*}^t, a_{u*}^t)/2)$;
      $e(terminal, x_u't) = d(s_{h*}^t, a_{u*}^t) + d(\alpha, a_{u*}^t) - d(s_{h*}^t, \alpha)/2$;
    end for
    for Algorithm 1 $m = 1 : M$ **do**
      $e(x_h^t, x_h^{t-1}) = f(\alpha, s_{h*}^{t-1}) + f(s_{h*}^t, \alpha) - f(s_{h*}^t, s_{h*}^{t-1})/2$;
      $e(terminal, x_h^t) = (-f(\alpha, s_{h*}^{t-1}) + f(s_{h*}^t, \alpha) + f(s_{h*}^t, s_{h*}^{t-1})/2) + (f(\alpha, s_{h*}^{t-1}) - f(s_{h*}^t, \alpha) + f(s_{h*}^t, s_{h*}^{t-1})/2) + (d(s_{h*}^t, a_{u*}^t) - d(\alpha, a_{u*}^t) + d(s_{h*}^t, \alpha)/2) + \lambda_{mh}$;
      $e(source, x_h^t) = \lambda_{\alpha h}$
    end for
  end for
end for
Solve the k-size s-t min cut [43] of $\mathcal{G} = (\mathcal{V}, \mathcal{E})$;

Algorithm 1: Auxiliary graph construction and solving algorithm.

$$
\begin{aligned}
&\Gamma_2' \min_{\mathcal{K},\mathcal{B}} \sum_{t=1}^{T} \sum_{m=1}^{M} \left(\varepsilon_1 \sum_{u=1}^{U} \frac{a_{mu}^t o_u^t}{B_{mu}^t \log_2\left(1 + \mathrm{SINR}_{mu}^t\right)} \varepsilon_2 \sum_{h=1}^{H} \frac{s_{mh}^t C_h^t}{k_{mh}^t} \right) \\
&\quad + \mathrm{Cost}_I + \sum_{t=1}^{T} \sum_{u=1}^{U} \Lambda_u^t \\
&C3' : \sum_{u \in U} B_{mu}^t \le B^t, \forall m \in \mathcal{M}, t \in \mathcal{T} \\
&C4' : \sum_{h \in H} k_{mh}^t \le K_m, \forall m \in \mathcal{M}, t \in \mathcal{T},
\end{aligned}
\tag{42}
$$

where Λ is penalty function, which can be expressed as

$$
\begin{aligned}
Z * \max \Bigg(& \sum_{h=1}^{H} p_u^h \Bigg[\left(\frac{a_{mu}^t o_u^t}{B_{u*}^t \log_2\left(1 + \mathrm{SINR}_{u*}^t\right)} + d\left(s_{h*}^t, a_{u*}^t\right) \right) \\
& + g\left(s_{h*}^t, s_{h*}^{t-1}\right) \Bigg) + \frac{s_{mh}^t C_h^t}{k_{h*}^t} \Bigg] - \mathcal{D}_u, 0 \Bigg),
\end{aligned}
\tag{43}
$$

where Z goes to infinity. $d(s_{h*}^t, a_{u*}^t)$ and $g(s_{h*}^t, s_{h*}^{t-1})$ are constants, when $\mathcal{A} *$ and $\mathcal{S} *$ are fixed.

Since the structure like $1/B_{mu}^t$ is a well-known convex function, the optimization problem can be proved to be a convex problem.

Since the variable k_{mh}^t can affect multiple spectrum allocation variables, we denote those as global variables. Next, the local copy of the global variables would be introduced. Each base station can obtain a distributed feasible solution by decoupling the above problem.

For BS m, we introduce the new variables $\widehat{\mathbf{k}}_m = \{\widehat{k}_{mh}^{et} | e \in \mathcal{M}, m \in \mathcal{M}, h \in \mathcal{H}, t \in \mathcal{T}\}$ as the local information.

$$
\widehat{k}_{mh}^{et} = k_{eh}^t, \forall e \in \mathcal{M}, m \in \mathcal{M}, h \in \mathcal{H}, t \in \mathcal{T}. \tag{44}
$$

$\widehat{\mathbf{B}}_m = \{\widehat{B}_{mu}^t | m \in \mathcal{M}, u \in \mathcal{U}\}$ is the local variation and represents the bandwidth resource allocation scheme of the BS m. Thus, the feasible local variables of the BS m can be denoted as $\Phi_m = (\widehat{\mathbf{k}}_m, \widehat{\mathbf{B}}_m)$ and the constraint set of the objective function can be denoted as Ω.

Let $\Psi(\Phi_m)$ be the penalty function, when the Φ_m belongs to the constraint set Ω, i.e.,$\Phi_m \in \Omega$, we can get $\Psi(\Phi_m) = 0$. Otherwise, $\Psi(\Phi_m) = +\infty$. So, the objective functions equivalent to

$$
\begin{aligned}
&\min_{\Phi_m}, \quad \sum_{m=1}^{M} \Xi_m(\Phi_m) + \Psi(\Phi_m) + \mathrm{Cost}_I \\
&\text{s.t.} \quad \widehat{k}_{mh}^{et} - k_{eh}^t = 0, \forall e \in \mathcal{M}, m \in \mathcal{M}, h \in \mathcal{H}, t \in \mathcal{T},
\end{aligned}
\tag{45}
$$

where $\Xi_m(\Phi_m) = \sum_{t=1}^{T} (\varepsilon_1 \sum_{u=1}^{U} (o_u^t / B_{mu}^t \log_2(1 + \mathrm{SINR}_{mu}^t)) + \varepsilon_2 \sum_{h=1}^{H} (C_h^t / k_{mh}^t))$, and in the above objective function, we can view Cost_I as a constant.

We separate the objective function into multiple local function of the corresponding BS. Each local function can determine its local variable by using local information. The Lagrange formula of the augmented problem is

$$\begin{aligned}\mathbb{L}\left(\{\Phi_m\}_{m\in\mathcal{M}}, \mathbf{k}, \{\xi_m\}_{m\in\mathcal{M}}\right) &= \sum_{m=1}^{M} \Xi_m(\Phi_m) + \Psi(\Phi_m) + Cost_I \\ &+ \sum_{m=1}^{M}\sum_{e=1}^{M}\sum_{h=1}^{H}\sum_{t=1}^{T} \xi_{mh}^{et}\left(\widehat{k}_{mh}^{et} - k_{eh}^{t}\right) \\ &+ \frac{\zeta}{2}\sum_{m=1}^{M}\sum_{e=1}^{M}\sum_{h=1}^{H}\sum_{t=1}^{T}\left(\widehat{k}_{mh}^{et} - k_{eh}^{t}\right)^2,\end{aligned} \tag{46}$$

where $\xi_m = \{\xi_{mh}^{et}\}$ are the vectors of the Lagrange multipliers, and the penalty parameter is $\zeta/2 \in \mathbb{R} + +$.

In order to solve the above problems (46), the iterative process is as follows.

4.3.1. Local Variables.

$$\begin{aligned}\Phi_m^{[\iota+1]} = \arg\min_{\Phi_m} &\Xi_m(\Phi_m) + \Psi(\Phi_m) + Cost_I \\ &+ \sum_{e=1}^{M}\sum_{h=1}^{H}\sum_{t=1}^{T} \xi_{mu}^{et[\iota]}\left(\widehat{k}_{mh}^{et} - k_{eh}^{t[\iota]}\right) \\ &+ \frac{\zeta}{2}\sum_{e=1}^{M}\sum_{h=1}^{H}\sum_{t=1}^{T}\left(\widehat{k}_{mh}^{et} - k_{eh}^{t[\iota]}\right)^2,\end{aligned} \tag{47}$$

where ι denotes the iteration times.

Since the updating process of Φ_m of each BS is independent, we can decouple the problem into M independent subproblems. We can update the local variables by solving the problem as follow:

$$\begin{aligned}\min_{\Phi_m} \quad & (\Phi_m) + \Psi(\Phi_m) + \mathrm{Cost}_I + \sum_{e=1}^{M}\sum_{h=1}^{H}\sum_{t=1}^{T} \xi_{mh}^{et[\iota]}\left(\widehat{k}_{mh}^{et} - k_{eh}^{t[\iota]}\right) + \frac{\zeta}{2}\sum_{e=1}^{M}\sum_{h=1}^{H}\sum_{t=1}^{T}\left(\widehat{k}_{mh}^{et} - k_{eh}^{t[\iota]}\right)^2 \\ \text{s.t.} \quad & \Phi_m \in \Omega.\end{aligned} \tag{48}$$

We solve the above problem by CVX, due to it being convex, and then, broadcast the decision of each BS to other BSs.

4.3.2. Global Variables.

$$\begin{aligned}\mathbf{k}^{[\iota+1]} = \arg\min_{k_{eh}} &\sum_{m=1}^{M}\sum_{e=1}^{M}\sum_{h=1}^{H}\sum_{T=1}^{T} \xi_{mh}^{et[\iota]}\left(\widehat{k}_{mh}^{et[\iota+1]} - k_{eh}^{t}\right) \\ &+ \frac{\zeta}{2}\sum_{m=1}^{M}\sum_{e=1}^{M}\sum_{h=1}^{H}\sum_{t=1}^{T}\left(\widehat{k}_{mh}^{et[\iota+1]} - k_{eh}^{t}\right)^2.\end{aligned} \tag{49}$$

The above problems are strictly convex and unconstrained quadratic problems, because we add the quadratic regular term to the augmented Lagrangian. Let the gradient of $\mathbf{k}$ be zero. We can get the following results:

$$\sum_{m=1}^{M} \xi_{mh}^{et[\iota]} + \zeta\sum_{m=1}^{M}\left(\widehat{k}_{mh}^{et[\iota+1]} - k_{eh}^{t}\right) = 0, \forall e, h, t. \tag{50}$$

And then, we can derive

$$k_{eh}^{t[\iota+1]} = \frac{1}{M\zeta}\sum_{m=1}^{M} \xi_{mh}^{et[\iota]} + \frac{1}{M}\sum_{m=1}^{M} \widehat{k}_{mh}^{et[\iota+1]}, \forall e, u, t. \tag{51}$$

By using $\sum_{m=1}^{M}\xi_{mh}^{et[\iota]} = 0$, we can derive

$$k_{eh}^{t[\iota+1]} = \frac{1}{M}\sum_{m=1}^{M} \widehat{k}_{mh}^{et[\iota+1]}, \forall e, u, t. \tag{52}$$

In other words, we can obtain global variables by averaging the corresponding updated local variables in each iteration.

4.3.3. Lagrange Multipliers.

$$\xi_m^{[\iota+1]} = \xi_m^{[\iota]} + \zeta\left(\widehat{\mathbf{k}}_m^{[\iota+1]} - \mathbf{k}^{[\iota+1]}\right). \tag{53}$$

At each iteration, we can calculate the Lagrange multipliers directly by using the updated local variables $\{\Phi_m\}$ and global variables $\{\mathbf{k}\}$. The formulation can be represented as follows:

$$\xi_{mh}^{et[\iota+1]} = \xi_{mh}^{et[\iota]} + \zeta\left(\widehat{k}_{mh}^{et[\iota+1]} - x_{eh}^{t[\iota+1]}\right). \tag{54}$$

4.3.4. Stopping Criterion and Convergence. The above problem is a convex problem with strong duality. When the number of iterations approaches infinity, the algorithm satisfies convergence. Therefore, the reasonable stopping criteria are given as follows:

$$\left\|\widehat{\mathbf{k}}_m^{[\iota+1]} - \mathbf{k}^{[\iota+1]}\right\|_2 \le \kappa_{\mathrm{pri}}, \forall m \in M, \tag{55}$$

$$\left\|\mathbf{k}^{[\iota+1]} - \mathbf{k}^{[\iota]}\right\|_2 \le \kappa_{\mathrm{dual}}, \forall m \in M, \tag{56}$$

where $\xi_{\mathrm{pri}} > 0$ and $\xi_{\mathrm{dual}} > 0$ indicate the primal feasibility and dual feasibility conditions, respectively, which are the small positive constant scalars.

The above iteration process based on convex optimization is concluded in Algorithm 2.

4.3.5. Two-Stage Iterative Algorithm Based on $\boldsymbol{\alpha}$ Expansion. Because there are many optimization variables in the original problem, the complexity of the algorithm is high. In order to reduce the algorithm complexity and obtain the

1: Initialization the number of iterations $\iota = 0$, global variables $\mathbf{k}^{[0]}$
and Lagrange multipliers $\xi^{[0]}$;
2: Set the maximum number of iterations $\iota_{\max}$ and the stopping criterion threshold ξ_{dual};
3: **while** $\iota < \iota_{\max}$, $\|\widehat{\mathbf{k}}_m^{[\iota+1]} - \mathbf{k}^{[\iota+1]}\|_2 > \kappa_{pri}$ and $\|\mathbf{k}^{[\iota+1]} - \mathbf{k}^{[\iota]}\|_2 > \kappa_{dual}$
4: Each BS m update Φ_m by solving problem (48), and share the local solution to other BSs;
5: Update the global variables $\mathbf{k}$ according to the formula (52);
6: Update the Lagrange multipliers ξ according to the formula (54);
7: $\iota = \iota + 1$;
8: **end while**
9: Output the optimal solution;

Algorithm 2: Resource allocation scheme based on convex optimization algorithm.

Input: $\mathcal{M}$ Set of BSs, $\mathcal{U}$ Set of players, $\mathcal{K}$ Set of groups, $\mathcal{T}$ Set of consecutive time slots;
Output: The variable $\mathcal{A}, \mathcal{S}, \mathcal{B}, \mathcal{K}$, and the minimum value of the objective function $Value_{best}$;
1: Initialization the variable s_{h*}^t = rand $[0, M]$, a_{u*}^t = rand $[0, M]$, k_{mh}^t, B_{mu}^t, and $Value_{best} = +\infty$
2: **for** iter=1:ι_3 **do**
3: **for** $\alpha \in \mathcal{M}, \sum_{h=1}^{H} \lambda_{\alpha h} s_{\alpha h} \le \Pi_\alpha$ **do**
4: run Algorithm 1, obtain $s_{h*}^{t\alpha}$, $a_{u*}^{t\alpha}$ and $Cost_I$
5: **for** iter=1:T **do**
6: run Algorithm 2, obtain k_{mh}^t, B_{mu}^t and $Value_{current}$
7: **end for**
8: **if** $Value_{current} < Value_{best}$ **then**
9: $Value_{bset} = Value_{current}$;
10: $s_{h*}^t = \alpha$, $a_{u*}^t = \alpha$;
11: **else**
12: $Value_{best} = Value_{best}$;
13: $s_{h*}^t = s_{h*}^t$, $a_{u*}^t = a_{u*}^t$;
14: **end if**
15: **end for**
16: **end for**

Algorithm 3: Two-stage iterative algorithm based on α expansion.

optimal solution to the original problem, we solve the original problem in two steps. So, we need to integrate the above two subalgorithms. Firstly, we input the result of Algorithm 1 as a fixed value into Algorithm 2 to solve Algorithm 2, and then, we compared the results of Algorithm 2 with the historical optimal results and updated the related variables. The above process is summarized in Algorithm 3.

Since we traverse for each MEC (Line 3 in Algorithm 3), the caching size can be restricted under Π_α at each round of α expansion.

4.4. Algorithm Complexity Analysis. Since Algorithms 1 and 2 are the modules invoked by Algorithm 3 for $M \times \iota_3$ times, where M is the number of MECs and ι_3 is the maximum number of iterations in Algorithm 3, we, respectively, investigate the complexity of Algorithms 1 and 2. According to paper [44], the complexity of the Algorithm 1 can be expressed as $\mathcal{O}(|\mathscr{E}||\mathscr{V}|^2)$, where $|\mathscr{V}|$ is the number of vertices and $|\mathscr{E}|$ is the number of edges in the constructed graph. In our case, $|\mathscr{V}| = T(U+H)+2$ is bounded by $\mathcal{O}(T(U+H))$; $|\mathscr{E}| = 3HUT + H(T+1) + TH$ is bounded by $\mathcal{O}(TUH)$.

Table 2: The simulation parameters.

Simulation parameters	Value
The total bandwidth	[0.8–1.2]GHz
The number of players	100
The number of BSs	10
The number of VR service modules	40
The downlink path loss exponent	[2.75–4.75]
The power spectral density of noise	-174 dBm/Hz
The transmission power of the players	0.1 W
The storage capability of the MEC	[400–900]G
The computing capability of the BS	[30–80]GHz

Therefore, the complexity of Algorithm 1 is $\mathcal{O}(T^3U^4)$, due to $H \le U$. For Algorithm 2, the variables a_{mu}^t and s_{mu}^t have been fixed, and the remaining question can be broken down into solving local optimization problem (48) at each BS by using ADMM algorithm, whose complexity is $\mathcal{O}(UH)$. $\iota_{\max}$ is the number of iterations required for Algorithm 2

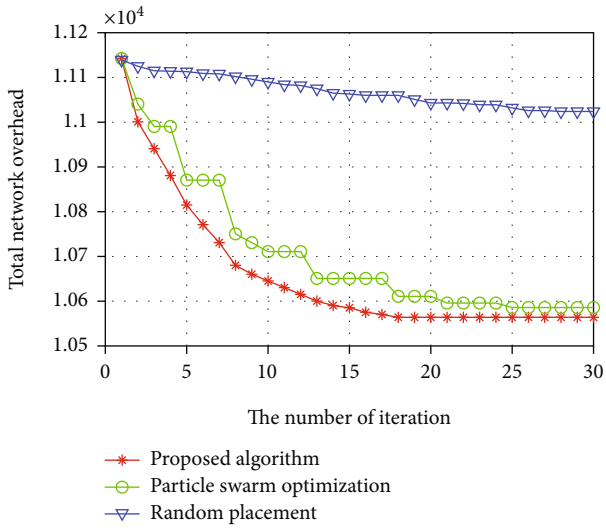


Figure 4: Total network overhead versus iteration times.

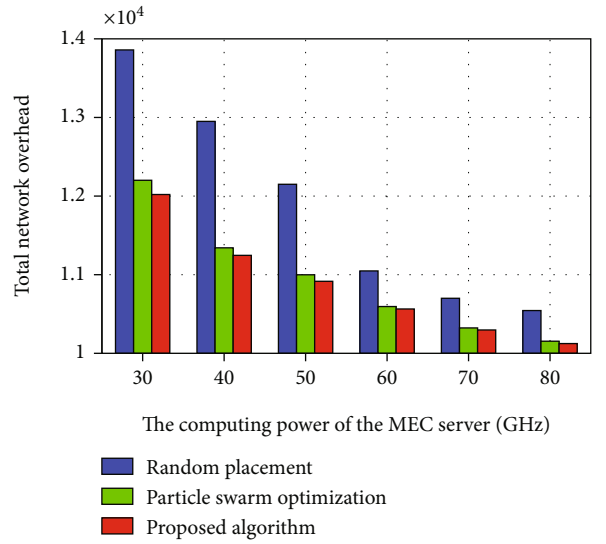


Figure 6: Total network overhead versus computing power of MEC server.

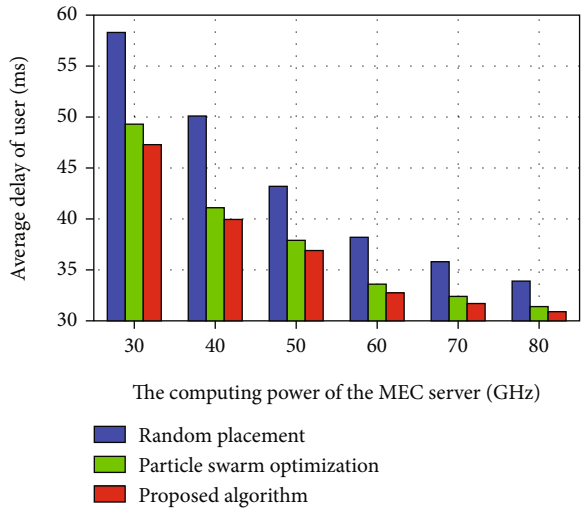


Figure 5: Average delay of user versus computing power of MEC server.

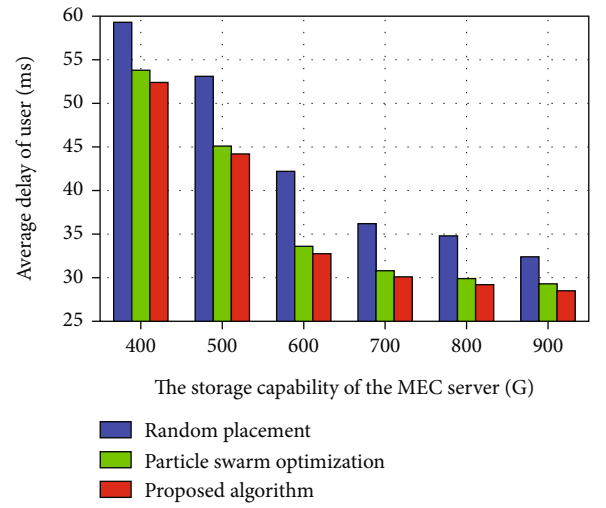


Figure 7: Average delay of user versus storage capacity of MEC server.

convergence; the total computational complexity is $\mathcal{O}(\iota_{\max} UH)$. Therefore, the overall complexity of Algorithm 3 is $\mathcal{O}(\iota_3 M(T^3U^4 + \iota_{\max} U^2 T))$.

5. Simulation Results and Discussions

In a wireless cellular network, it is assumed that 100 players and 10 base stations are randomly distributed in a circle with a radius of 100 m; other major simulation parameters are shown in Table 2.

To evaluate the performance of our proposed approach, we compare our proposed α expansion-based two-stage approach to two other approaches: (1) placing each VR service module randomly on a MEC at each time slot, as labeled as "random placement," and (2) particle swarm optimization was used to solve the objective function, as labeled as "particle swarm optimization."

In Figure 4, we iteratively find the minimum value of the total network overhead under the condition that the maximizing computing capacity of each MEC server is 60 GHz and the maximizing storage capacity of each MEC server is 600 G, where total network overhead is the sum of the adjusted placement cost, communication cost, migration cost, and rendering cost, i.e., this paper's object function $\sum_{t=1}^{T} \varepsilon_1 \text{Cost}_C^t + \varepsilon_2 \text{Cost}_R^t + \varepsilon_3 \text{Cost}_P^t + \varepsilon_4 \text{Cost}_M^t$. As shown above, the total network overhead of our proposed scheme and particle swarm optimization decreases rapidly as the iteration increases at the beginning, and then, the total network overhead converges and remains at an almost constant value. Moreover, it can be seen from the iteration diagram that our proposed algorithm converges in about 18 generations, while particle swarm

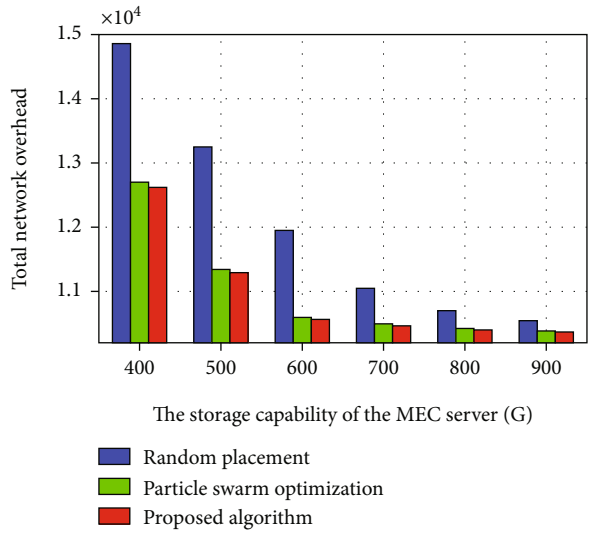


Figure 8: Total network overhead versus storage capacity of MEC server.

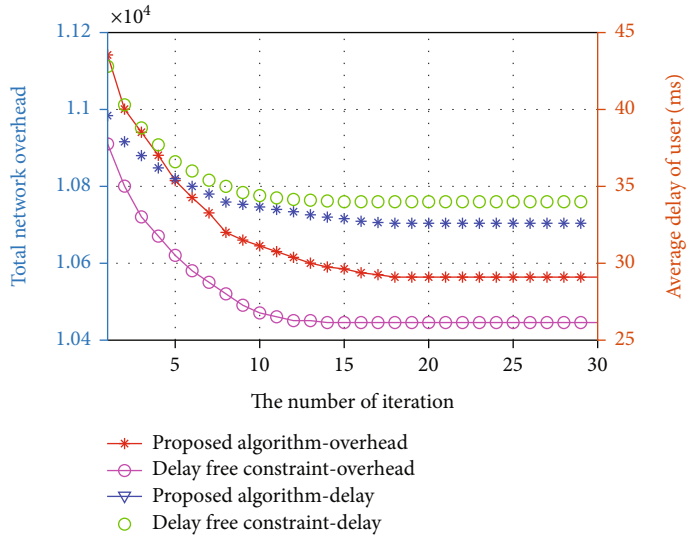


Figure 9: The influence of delay constraint on total network overhead and average delay of user.

optimization converges in about 25 generations. So, compared with other schemes, our proposed algorithm converges faster in the iteration process and keeps the lowest total network overhead.

Figure 5 shows the relationship between the computing power of the MEC server and the user average latency. Figure 6 shows the relationship between the computing power of the MEC server and the total network overhead. In the above two figures, as the computing power of the MEC server increases, the average latency and total network overhead of the user are greatly reduced. This is mainly because the more computing resources a MEC server can provide to the player, the less latency it needs to perform rendering. At the same time, the richer computing resources on the MEC server, the more MEC servers the system could be chosen to provide rendering services for a group of VR players, which saves the network cost of routing.

Figure 7 shows the relationship between the storage capability of the MEC server and the user average latency. Figure 8 shows the relationship between the storage capability of the MEC server and the total network overhead. As shown in the above two figures, the placement strategy proposed by us can effectively reduce the total network overhead. Moreover, with the storage capacity of the MEC server increasing, the average latency of user and total network overhead is greatly reduced. This is mainly because the larger the storage capacity of the MEC server, the more VR service modules can be placed on each edge node, which can reduce the migration costs between two base stations to a certain extent. Especially when the number of VR service modules that can be placed on the MEC server is small, in order to meet the video processing requirements of the constantly moving player, VR service modules need to migrate frequently between base stations. As shown in Figure 8, when

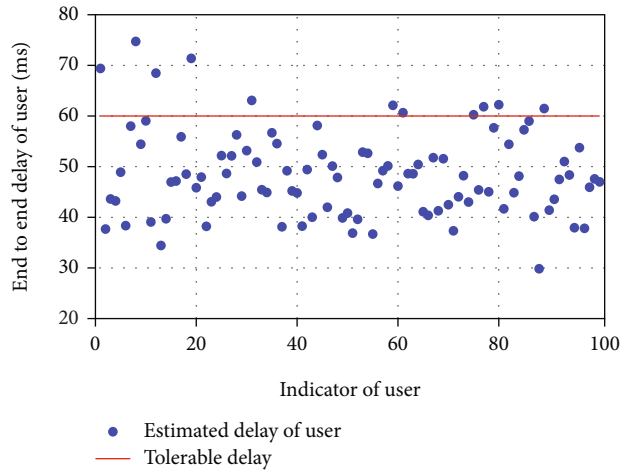


Figure 10: The actual delay of each player under the condition of no delay constraint and the tolerable delay.

the storage capability of the MEC server is less than 600 G, VR service module migration between base stations becomes frequent, and the total network overhead increases greatly.

In Figure 9, we compare the user average delay and total network overhead without delay constraint with the user average delay and total network overhead with delay constraint. The network parameter is the maximizing computing capacity of each MEC server is 60 GHz, and the storage capacity of each MEC server is 600 G. When there is no need to consider satisfying the delay constraint of each user, the feasible domain of the target problem becomes larger, and the total network cost is reduced compared with when the delay constraint is considered, but the average delay of the user will increase. At the same time, some users cannot complete their corresponding video processing tasks within the tolerable delay, as shown in Figure 10.

6. Conclusion

In this paper, we develop dynamical service module placement strategies based on rendering-aware to minimize the sum of the network costs over a long time under satisfying the delay constraint of each player. The strategies jointly consider the resource allocation scheme within each time slot and the service module migration scheme between different base stations in the adjacent time slot. Moreover, we propose a two-stage algorithm based on graph cut and convex optimization to solve the objective function. In future work, we will study the online placement strategy of VR service modules to further improve user experience and reduce network overhead in the process of VR video stream delivery and computing. In addition, we will extend our work to the security [45] and low-delay delivery of all kinds of superlarge video streams.

Data Availability

The simulation data used to support the findings of this study are included in the article. The research status data used to support the findings of this study are available in the references of this article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (No. 62171061).

References

[1] L. Wang, L. Jiao, T. He, J. Li, and H. Bal, "Service placement for collaborative edge applications," *IEEE/ACM Transactions on Networking*, vol. 29, no. 1, pp. 34–47, 2021.

[2] R. Fantacci and B. Picano, "End-to-end delay bound for wireless uVR services over 6G terahertz communications," *IEEE Internet of Things Journal*, vol. 8, no. 23, pp. 17090–17099, 2021.

[3] P. Bhattacharya, D. Saraswat, A. Dave et al., "Coalition of 6G and blockchain in AR/VR space: challenges and future directions," *IEEE Access*, vol. 9, pp. 168455–168484, 2021.

[4] P. Lin, Q. Song, F. R. Yu, D. Wang, and L. Guo, "Task offloading for wireless VR-enabled medical treatment with blockchain security using collective reinforcement learning," *IEEE Internet of Things Journal*, vol. 8, no. 21, pp. 15749–15761, 2021.

[5] C. Zheng, S. Liu, Y. Huang, and L. Yang, "Hybrid policy learning for energy-latency tradeoff in MEC-assisted VR video service," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 9, pp. 9006–9021, 2021.

[6] Y. Cao, R. Ji, L. Ji, G. Lei, H. Wang, and X. Shao, "l^2-MPTCP: a learning-driven latency-aware multipath transport scheme for industrial internet applications," *IEEE Transactions on Industrial Informatics*, 2022.

[7] G. Lei, L. Ji, R. Ji, Y. Cao, and X. Huang, "Extracting low-rate DDoS attack characteristics: the case of multipath TCP-based communication networks," *Wireless Communications and Mobile Computing*, vol. 2021, no. 4, Article ID 2264187, p. 10, 2021.

[8] L. Qin, Y. Cao, X. Shao et al., "A deep heterogeneous optimization framework for Bayesian compressive sensing," *Computer Communications*, vol. 178, pp. 74–82, 2021.

[9] Y. Ye, R. Q. Hu, G. Lu, and L. Shi, "Enhance latency-constrained computation in mec networks using uplink Noma," *IEEE Transactions on Communications*, vol. 68, no. 4, pp. 2409–2425, 2020.

[10] X. Liu and Y. Deng, "A decoupled learning strategy for mec-enabled wireless virtual reality (vr) network," in *2021 IEEE International Conference on Communications Workshops (ICC Workshops)*, pp. 1–6, Montreal, QC, Canada, 2021.

[11] J. Du, F. R. Yu, G. Lu, J. Wang, J. Jiang, and X. Chu, "MEC-assisted immersive vr video streaming over terahertz wireless networks: a deep reinforcement learning approach," *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 9517–9529, 2020.

[12] L. Wang, L. Jiao, J. Li, J. Gedeon, and M. Muhlhauser, "MOERA: mobility-agnostic online resource allocation for edge computing," *IEEE Transactions on Mobile Computing*, vol. 18, no. 8, pp. 1843–1856, 2019.

[13] H. Wu, L. Liu, X. Zhang, and H. Ma, "Vbargain: a market-driven quality oriented incentive for mobile video offloading," *IEEE Transactions on Mobile Computing*, vol. 18, no. 9, pp. 2203–2216, 2019.

[14] X. Shao, G. Hasegawa, M. Dong, Z. Liu, H. Masui, and Y. Ji, "An online orchestration mechanism for general-purpose edge computing," *IEEE Transactions on Services Computing*, pp. 1–1, 2022.

[15] I. Labriji, F. Meneghello, D. Cecchinato et al., "Mobility aware and dynamic migration of MEC services for the internet of vehicles," *IEEE Transactions on Network and Service Management*, vol. 18, no. 1, pp. 570–584, 2021.

[16] F. Guo, F. R. Yu, H. Zhang, H. Ji, V. C. M. Leung, and X. Li, "An adaptive wireless virtual reality framework in future wireless networks: a distributed learning approach," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 8, pp. 8514–8528, 2020.

[17] A. Yousafzai, I. Yaqoob, M. Imran, A. Gani, and R. Md Noor, "Process migration-based computational offloading framework for IoT-supported mobile edge/cloud computing," *IEEE Internet of Things Journal*, vol. 7, no. 5, pp. 4171–4182, 2020.

[18] J. Feng, F. R. Yu, Q. Pei, J. Du, and L. Zhu, "Joint optimization of radio and computational resources allocation in blockchain-enabled mobile edge computing systems," *IEEE Transactions on Wireless Communications*, vol. 19, no. 6, pp. 4321–4334, 2020.

[19] J. Liu and Q. Zhang, "Reliability and latency aware code-partitioning offloading in mobile edge computing," in *2019 IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1–7, Marrakesh, Morocco, 2019.

[20] L. Pu, L. Jiao, X. Chen, L. Wang, Q. Xie, and J. Xu, "Online resource allocation, content placement and request routing for cost-efficient edge caching in cloud radio access networks," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 8, pp. 1751–1767, 2018.

[21] L. Wang, L. Jiao, T. He, J. Li, and M. Muhlhauser, "Service entity placement for social virtual reality applications in edge computing," in *IEEE INFOCOM 2018- IEEE Conference on Computer Communications*, pp. 468–476, Honolulu, HI, USA, 2018.

[22] L. Guo, J. Pang, and A. Walid, "Joint placement and routing of network function chains in data centers," in *IEEE INFOCOM 2018 – IEEE Conference on Computer Communications*, pp. 612–620, Honolulu, HI, USA, 2018.

[23] P.-Y. Chou, W.-Y. Chen, C.-Y. Wang, R.-H. Hwang, and W.-T. Chen, "Deep reinforcement learning for mec streaming with joint user association and resource management," in *ICC 2020-2020 IEEE International Conference on Communications (ICC)*, pp. 1–7, Dublin, Ireland, 2020.

[24] V. Farhadi, F. Mehmeti, T. He et al., "Service placement and request scheduling for dataintensive applications in edge clouds," in *IEEE INFOCOM 2019 – IEEE Conference on Computer Communications*, pp. 1279–1287, Paris, France, 2019.

[25] Y. Han, D. Guo, W. Cai, X. Wang, and V. Leung, "Virtual machine placement optimization in mobile cloud gaming through QoE-oriented resource competition," *IEEE Transactions on Cloud Computing*, pp. 1–1, 2020.

[26] X. Shao, H. Asaeda, M. Dong, and Z. Ma, "Cooperative inter-domain cache sharing for information-centric networking via a bargaining game approach," *IEEE Transactions on Network Science and Engineering*, vol. 6, no. 4, pp. 698–710, 2019.

[27] Y. Liu, J. Liu, A. Argyriou, L. Wang, and Z. Xu, "Rendering-aware VR video caching over multi-cell MEC networks," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 3, pp. 2728–2742, 2021.

[28] J. Dai, Z. Zhang, S. Mao, and D. Liu, "A view synthesis-based 360° VR caching system over MEC-enabled C-RAN," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 10, pp. 3843–3855, 2020.

[29] Y. Ye, L. Shi, X. Chu, R. Q. Hu, and G. Lu, "Resource allocation in backscatter-assisted wireless powered MEC networks with limited MEC computation capacity," *IEEE Transactions on Wireless Communications*, pp. 1–1, 2022.

[30] W. Lu, X. Meng, and G. Guo, "Fast service migration method based on virtual machine technology for MEC," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4344–4354, 2019.

[31] S. Wang, J. Xu, N. Zhang, and Y. Liu, "A survey on service migration in mobile edge computing," *IEEE Access*, vol. 6, no. 23, pp. 23511–23528, 2018.

[32] L. Yang, D. Yang, J. Cao, Y. Sahni, and X. Xu, "QoS guaranteed resource allocation for live virtual machine migration in edge clouds," *IEEE Access*, vol. 8, pp. 78441–78451, 2020.

[33] L. Liang, J. Xiao, Z. Ren, Z. Chen, and Y. Jia, "Particle swarm based service migration scheme in the edge computing environment," *IEEE Access*, vol. 8, pp. 45596–45606, 2020.

[34] P. Fang, Y. Zhao, Z. Liu, J. Gao, and Z. Chen, "Resource allocation strategy for MEC system based on vm migration and rf energy harvesting," in *2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring)*, pp. 1–6, Antwerp, Belgium, 2020.

[35] K. Liu, J. Peng, J. Wang, W. Liu, Z. Huang, and J. Pan, "Scalable and adaptive data replica placement for geo-distributed cloud storages," *IEEE Transactions on Parallel and Distributed Systems*, vol. 31, no. 7, pp. 1575–1587, 2020.

[36] K. Poularakis, J. Llorca, A. M. Tulino, I. Taylor, and L. Tassiulas, "Joint service placement and request routing in multi-cell mobile edge computing networks," in *IEEE INFOCOM 2019- IEEE Conference on Computer Communications*, pp. 10–18, Paris, France, 2019.

[37] S. Li, P. Lin, J. Song, and Q. Song, "Computing-assisted task offloading and resource allocation for wireless VR systems," in *2020 IEEE 6th International Conference on Computer and Communications (ICCC)*, pp. 368–372, Chengdu, China, 2020.

[38] C. Zheng, S. Liu, Y. Huang, and L. Yang, "MEC-enabled wireless VR video service: a learning-based mixed strategy for energy-latency tradeoff," in *2020 IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1–6, Seoul, Korea (South), 2020.

[39] H. Xiao, C. Xu, Z. Feng et al., "A transcoding-enabled 360° VR video caching and delivery framework for edge-enhanced next-generation wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 40, no. 5, pp. 1615–1631, 2022.

[40] K. Long, Y. Cui, C. Ye, and Z. Liu, "Optimal wireless streaming of multi-quality 360 VR video by exploiting natural, relative smoothness-enabled, and transcoding-enabled multicast opportunities," *IEEE Transactions on Multimedia*, vol. 23, pp. 3670–3683, 2021.

[41] T. Bai and R. W. Heath, "Coverage and rate analysis for millimeter-wave cellular networks," *IEEE Transactions on Wireless Communications*, vol. 14, no. 2, pp. 1100–1114, 2015.

[42] S. Singh, M. N. Kulkarni, A. Ghosh, and J. G. Andrews, "Tractable model for rate in self-backhauled millimeter wave cellular

networks," *IEEE Journal on Selected Areas in Communications*, vol. 33, no. 10, pp. 2196–2211, 2015.

[43] P. Zhang, "A new approximation algorithm for the unbalanced min *s-t* cut problem," *Theoretical Computer Science*, vol. 609, pp. 658–665, 2016.

[44] J. R. Edmonds and R. M. Karp, "Theoretical improvements in algorithmic efficiency for network flow problems," *Journal of the ACM (JACM)*, vol. 19, no. 2, article 248C264, pp. 248–264, 1972.

[45] Y. Cao, R. Ji, L. Ji, X. Shao, G. Lei, and H. Wang, "MPTCP-*me*Learning: a multi-expert learning-based MPTCP extension to enhance multipathing robustness against network attacks," *Transactions on Information and Systems*, vol. E104.D, no. 11, pp. 1795–1804, 2021.

Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 6300715, 14 pages
https://doi.org/10.1155/2022/6300715

Research Article

Decentralized Vehicular Mobility Management Study for 5G Identifier/Locator Split Networks

Gaofeng Hong,[1] **Bin Yang**,[2] **Wei Su**,[1] **Qili Wen**,[1] **Xindi Hou**,[1] **and Haoru Li**[1]

[1]*School of Electronic and Information Engineering, Beijing Jiaotong University, China*
[2]*School of Computer and Information Engineering, Chuzhou University, China*

Correspondence should be addressed to Gaofeng Hong; honggf@bjtu.edu.cn

Received 8 April 2022; Accepted 23 June 2022; Published 15 July 2022

Academic Editor: Yuanlong Cao

The identifier/locator split (ILS) architectures are highly promising to reduce the signaling latency of frequent handovers in fifth generation (5G) networks, while decentralized vehicular mobility management holds greater potential than the traditional centralized management to enhance the critical performance of highly dynamic and dense cell networks. By carefully exploiting ILS, dual connectivity, and multiaccess edge computing (MEC) concepts, this paper proposes a decentralized vehicular mobility management mechanism in the network with dense 5G Non-Standalone deployment. Under such a mechanism, we design an ILS-based local anchor handover management architecture to reduce signaling costs and handover latency. Specifically, we propose a quality of service- (QoS-) based handover decision algorithm using a long short-term memory- (LSTM-) based trajectory prediction method to obtain the cell sojourn time of connected vehicles (CVs) in predefined QoS coverage areas. Combining a built-in dynamic handover trigger condition, this algorithm can ensure a flexible load balance as well as low handover times. Extensive simulation results are presented to verify the effectiveness of the proposed mechanism in improving network performance.

1. Introduction

The ILS architectures, which decouple the IP address semantics into two types of roles, namely, user equipment (UE), access identifiers (AIDs), and routing identifier (RIDs), have been identified as a promising paradigm for the 5G and beyond wireless networks [1]. Such architectures can significantly reduce the cost of frequent handovers in highly dynamic networks [2]. For instance, because of the dense gNB deployment and fast moving vehicles, handovers occur frequently in the emerging 5G vehicular networks [3–5], which can cause heavy handover signaling load in traditional network architectures with the overloading of IP address semantics. Fortunately, ILS architectures have the potential to guarantee vehicular communications without occurring outage [6]. However, vehicular communications based on the ILS architecture still face some fundamental challenges in the 5G dense cell scenarios, like mapping management and real-time handover decisions [7]. Therefore, it is critical to explore a novel vehicular mobility management mechanism for improving the quality of the vehicular communications.

Existing ILS architectures, such as lisp mobile node (LISP-MN) [8], Host Identity Protocol (HIP) [9], MILSA [10], MobilityFirst [11], and Smart Identifier Network (SINET) [12], provide various optimization schemes of the mobility management. Relevant schemes can be classified into two categories, namely, mobility signaling [13, 14] and mapping system update efficiency [15, 16]. The former aims at simplifying the handover signaling interaction to support a seamless roaming, while the latter is to boost the efficiency of mapping update to avoid the outdated identifier-to-locator mapping during a new handover.

These works mainly focus on the centralized mobility management system with relatively low handover frequency and simple handover decision. Recently, some initial works

focus on the highly dynamic 5G ILS networks [6, 17], which are intended to provide reliable communications of high-speed railways.

It is notable that all above works demonstrate the potentials of ILS architectures in reducing the mobility management costs. However, these results cannot be directly applied to the 5G vehicular networks. On one hand, the frequent handovers in such networks can bring heavy update load to the centralized mapping system via the domain gateway (GW), which may lead to a single point of failure once if the requirements of vehicular communications increase. On the other hand, the moving routes of vehicles are more complex than high-speed trains, and thus the handover decisions proposed in high-speed railway communications cannot be directly applied to the complex target access network selection of the vehicular communications. Furthermore, the handover decision algorithms have also been proven to play vitally important roles in reducing the number of handovers and network load imbalance among the access entities [18]. Specifically, lacking efficient handover decision algorithms could not only degrade the Quality of Service (QoS) but also negate the benefits of mobility management using the ILS architecture in the 5G vehicular networks.

By now, relevant research efforts have illustrated the superiority of decentralized mobility management in the traditional centralized cellular networks [19–24]. In addition, various handover decision algorithms in such networks are also proposed in [25–28] (please see Related Works of Section 2 for details). A fundamental issue is how to explore a distributed ILS mapping management scheme with an optimal handover decision algorithm, which fully considers UE's requirements and network state in the dense 5G scenario. However, this issue has not been well addressed by now. Motivated by this observation, the paper presents a decentralized 5G vehicular mobility management architecture, which is based on the 5G Non-Standalone (NSA) dual connectivity (DC) networking [29–31] and the decentralized management capacity of the MEC technology [32]. It aims to proactively detect, predict, and perform fast handovers by fully exploiting the advantages of the ILS architecture in the 5G vehicular networks. To ensure the performance of the proposed mobility management architecture, we carefully address an efficient handover decision algorithm based on the movement characteristic of vehicles and the requirements of on-board services, which can largely reduce the overhead of mobility management and the unnecessary handover times as well as increasing the utilization of network resource.

This paper extends our previous work [33], and the main contributions can be summarized as follows:

(i) We propose a Local Mobility Anchor- (LMA-) based handover management architecture in the Evolved Packet Core- (EPC-) based 5G NSA networking mode reusing the current LTE facilities. DC technology in such an architecture, the LTE master eNB (MeNB), which serves as an LMA, takes charge of the control-plane (*C*-plane) procedures and is also a backup for the user-plane (*U*-plane) transmission of the 5G secondary gNB (SgNB). Remarkably, each MeMB is attached to an MEC server, which can process handover context parameter, manage local mapping information, and execute the predefined handover decision algorithm in a distributed manner. We further give the collection and management methods of the context information and the fast handover procedures for intra-MeNB and inter-MeNB handovers

(ii) For the intra-MeNB handover, we further propose a vehicular handover decision algorithm with the aim of reducing the number of SgNB handovers, balancing SgNB load and satisfying different networks requirements of CV services. In this algorithm, we design a novel QoS coverage conversion method to determine the QoS boundary of candidate gNBs including the specific service requirement and the real-time network load. Based on this method, we develop an LSTM-based trajectory prediction model, which is used to determine a vital decision variable, i.e., the sojourn time of a vehicle residing in the QoS boundary of each candidate gNB

(iii) Based on the predicted sojourn time and the real-time network status, we redefine the trigger condition of the intra-MeNB handover as a dynamic Time-to-Trigger (TTT) value, which enhances the robustness of mobility management in highly dynamic handover context due to the heterogeneity of 5G gNBs

(iv) Extensive simulation results are presented to validate the prediction accuracy of the trajectory prediction model in our proposed mechanism and also to conduct a comparison between our proposed mechanism and a classic traditional one

The rest of the paper is organized as follows. Section 2 presents the related works. Section 3 introduces our concerned network model. In Section 4, we propose the LMA-based handover management architecture and the relative handover procedures under this architecture. Section 5 further gives the QoS-based handover decision algorithm. Extensive simulation results are provided in Section 6. Finally, Section 7 concludes this paper. The abbreviations used in this paper are provided in Table 1.

2. Related Works

2.1. Distributed Mobility Management Mechanisms. The Distributed Mobility Management (DMM) mechanisms [34, 35] distribute the control and data functions among several infrastructures located at the edge of the network, instead of relying on a single central server in traditional centralized network. The DMM mechanisms are proposed in distributed ILS-based mobile networks to relieve the signaling loads and handover delays [19, 20]. The potentials of such mechanisms are further shown in cellular networks [21]. The work in [22] proposes an efficient local mobility

Table 1: Key abbreviations.

Abbreviation	Description
ILS	Identifier/locator split
AID	Access identifier
RID	Routing identifier
GAID	Global access identifier
LAID	Local access identifier
QoS	Quality of service
GW	Domain gateway
CV	Connected vehicle
MeNB	Master eNodeB (LTE base station)
SgNB	Secondary gNodeB (5G base station)
EPC	Evolved Packet Core
DC	Dual connectivity
LMA	Local Mobility Anchor
LSD	Location service domain
NSA	Nonstandalone networking
C-plane	Control-plane (used for the interactive control signaling between the user and the network)
U-plane	User-plane (used for data traffic transmission of users)
MEC	Multiaccess edge computing technology
LSTM	Long short-term memory (a variant of recurrent neural network)
DMM	Distributed Mobility Management
RSU	Road side unit
OBU	On-board unit
CAM	Cooperative awareness message
RSRP	Reference signal receiving power
RRC	Radio Resource Control
TN	Target node
RLF	Radio link failure
TTT	Time-to-Trigger (the handover is initiated only if the triggering requirement is fulfilled for a certain time interval, which is called TTT)

management mechanism in a dense cell scenario. Under the mechanism, once if a handover happens, the target cell can establish a local path based on the X2 interface with the serving cell without sending a handover request to the core network. Similarly, two finer granularity location management mechanisms are proposed in dense cell networks [23], where the UE's location in a cell or a tracking area is registered to an LMA selected by the surrounding cells, and then the signaling of location update is transmitted using the X2 interface for reducing the overhead at the core network. However, the efficiency of such DMM schemes probably depends on the network service duration, compared to the UE sojourn time within the coverage of the cells [24]. The DMM mechanisms also show their limitations in their performance when the UE is in a high-speed state and the cell sojourn time becomes shorter. Therefore, another critical issue is to design an appropriate handover decision algorithm for improving the performance of DMM mechanisms.

2.2. Handover Decision Algorithms. The work in [36] illustrates that a proper handover decision algorithm can significantly mitigate the negative impact of UE's mobility on the QoS. The impact of user trajectories on the final handover decision is analyzed by deriving the closed-form expressions for the relative mobility model and the handover rate [25]. The authors in [26] propose a mobility state estimation algorithm, with which UEs are divided into different classes based on their velocity, and each class is associated with a handover trigger condition to minimize their handover failure rate. These results of [25, 26] are still not well applied to practical scenarios, where the actual vehicular trajectory is more complicated such that an accurate cell sojourn time is difficult to be obtained. In [27], a trajectory prediction algorithm based on deep learning has been used in handover decisions of heterogeneous vehicular networks. The predication algorithm can effectively improve the accuracy of mobility prediction and reduce unnecessary handovers. Meanwhile, the Cell Range Expansion technique in [28] is utilized to appropriately enlarge the small cell coverage to control the number of UEs access in a specific cell, which relieve the imbalance network load brought by frequent handovers.

3. Network Architecture

As shown in Figure 1, we present an ILS-based 5G network architecture consisting of five main communication entities, namely, LTE MeNB, 5G secondary gNB (SgNB), MEC server, GW, road side unit (RSU), and connected vehicle (CV), which is based on the EPC-based 5G NSA DC networking. The functions of these entities are introduced as follows:

MeNB: it can provide radio coverage over a larger area, which is responsible for both *C*-plane and *U*-plane transmission, working as a mobility anchor for the SgNBs. Here, the *U*-plane transmission is used only when no SgNB is available, and the control region of a MeNB is also called a location service domain (LSD)

SgNB: it can cover a relatively small area, which is responsible for user plane transmission, enhancing system capacity and providing high data transmission rate for vehicles

MEC server: it is placed near MeNBs and serves as a distributed local mapping server, which is responsible for handover context information management, executing the optimal network selection algorithm to make handover decisions

GW: in addition to acting as a gateway between MeNB and the Internet, it manages mapping information between CVs and each LSD

RSU: it obtains the driving status information of vehicles within its coverage area, and then sends the information to MEC servers through fiber line

CV: it is equipped with multiple types of communication modules. The on-board unit (OBU) periodically broadcasts Cooperative Awareness Messages (CAMs) [37] such that RSU can receive CV's real-time motion status through the Cellular-Vehicle-to-Everything (C-V2X) technology. Meanwhile, it maintains uninterrupted communication with

cellular base stations and requests varieties of vehicular services from the remote server through cellular network connection

The CV in this network architecture connects to MeNB and SgNB simultaneously based on the DC technology. The MeNB acts as a mobility anchor for *C*-plane transmission and a backup for *U*-plane transmission. The SgNB does not exchange control signals with the core network but enhances *U*-plane transmission. In this paper, we assume that the MEC server is merged with an MeNB, which provides extra storage and computing capacity of the MeNB for the subsequent mobility management. The combination of an MeNB and an MEC server is regarded as a mobility anchor.

4. Decentralized LMA-Based Handover Management Architecture

A decentralized LMA-based handover management architecture is proposed in this section. Figure 2 illustrates the relative functions and behaviors under this architecture consisting of the obtaining and management of the handover context parameters, the hierarchical mapping system, the optimal handover decision algorithm, and the handover executing procedure (intra-MeNB or inter-MeNB). The details of the optimal handover decision algorithm will be depicted in the next section.

4.1. Obtaining and Management of Context Parameters. To make optimal handover decision, it is essential to obtain the context parameters like SgNB coverage size, SgNB traffic loads, and vehicle motion trajectory. The MEC server associated with the MeNB is responsible for managing the context parameters of the SgNBs and CVs under the coverage of the MeNB. These parameters can be divided into the following three categories:

4.1.1. Cooperative Awareness Message. There is the driving status of the CV v_k in the CAM defined by the ETSI standard [37]. The CV periodically transmits CAMs to RSUs. To reduce the transmission cost, each CAM only contains PDU header, basic container, and HF container. When a RSU receives the CAM, it will synchronize the driving status information with the time stamp to the specified MEC server through fiber lines.

4.1.2. Measurement Report. The Radio Resource Control (RRC) connection can be established between the CV and the MeNB. The MeNB provides measurement configuration to the CV through RRC connection reconfiguration [38]. The measurement configuration includes candidate SgNBs, measurement parameters, and measurement period. Here, the measurement parameters consist of reference signal receiving quality, current received data transmission rate, and service request data rate. The CV measures the link quality of the nearby networks according to the measurement configuration in the RRC connection reconfiguration message and periodically uploads measurement reports to the MeNB. The MeNB will synchronize the measurement report to the MEC server.

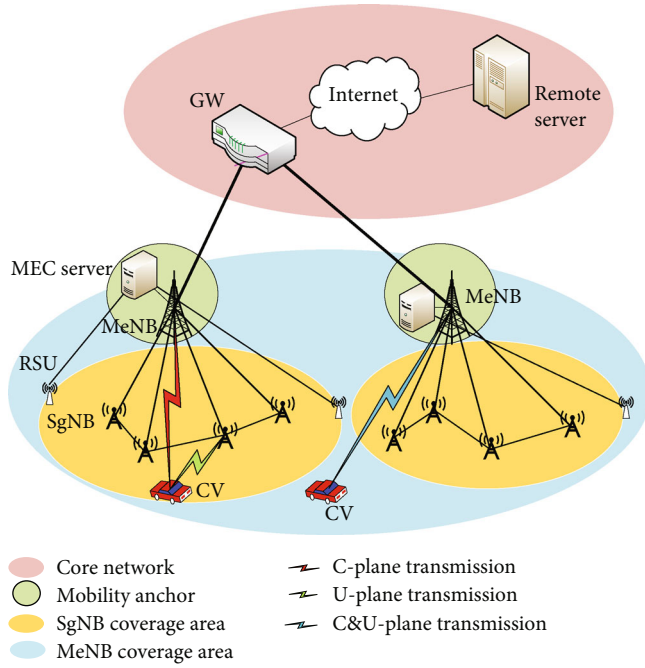


Figure 1: ILS-based LTE-5G dual connectivity network architecture.

4.1.3. Interaction between MeNBs and SgNBs. The MeNB, which serves as a mobility anchor, maintains the context information of its SgNBs, in terms of the maximum transmission power, residual available bandwidth, the number of current connected terminals, etc. These information will be periodically synchronized to the MEC server through C-plane transmission, with negligible transmission delay.

Based on the first two categories of monitoring methods, we use a CV context set $C^t_{v_k}$ to synchronize the driving state and the on-board service network requirement of v_k in the time slot t. The driving state consists of velocity, historical position sequence, acceleration, etc.

A context information table (CIT) between CVs and candidate SgNBs can be built and updated on the MEC server, as depicted in Table 2.

4.2. Identifier-Locator Mapping System. In the proposed architecture, each CV is identified by a global AID (GAID) and a local AID (LAID). The GAID is unique and represents a CV's identity in the global DNS system. The LAID only exists when a CV is attached to a SgNB. The dynamic LAID is highly related to the RID of the SgNB associated with the CV. Meanwhile, each SgNB and MeMB have an RID serving as a global service location inside the core network and can be globally routing.

Each GW has a set of GAID_{CV}-to-RID_{MeNB} mapping cache entries, which is used to map each CV to its MeNB LSD.

Besides, each MEC server on the MeNB side has three sets of mapping cache entries: GAID_{CV}-to-RID_{SgNB}, RI

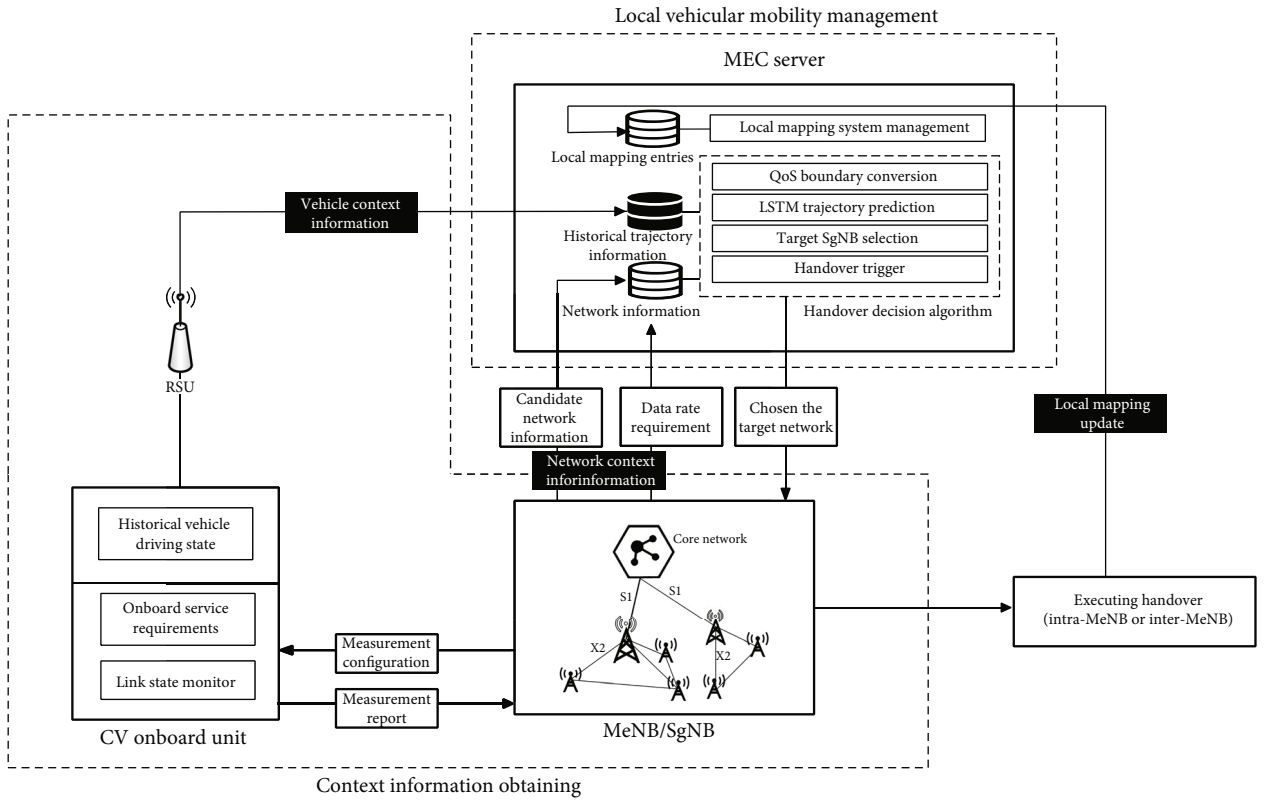


Figure 2: The LMA-based handover management architecture.

D_{SgNB}-to-RID_{MeNB}, and $GAID_{TN}$-to-RID_{MeNB}. The first entry maps the identifier of the CV to its connected SgNB service area, and the second entry maps SgNB global location to its local anchor, while the last entry is used to map the identifier of the target communication node to the current MeNB LSD through the core network.

4.3. Intra-MeNB Handover Procedure. When an intra-MeNB handover (SgNB-to-SgNB) is triggered by the MEC server, the CV will prepare an air interface to turn on for the target SgNB and continuously monitor the link status. Meanwhile, CV sends a Map Select Request message to the attached MeNB to notify its movement, which contains its $GAID_{TN}$ list and old and new LAIDs. MeNB configures a new LAID for the CV and starts a fast intra handover by sending Map Forwarding Request message to the target SgNB, which applies the access admission for the CV. The target SgNB sends a Map Forwarding Response to MeNB, and then the previous transmitting data is forwarded to the target SgNB. In this process, the MeNB acts as a backup U-plane transmission to CV, keeping a zero interruption data forwarding. Then, the MeNB responds to CV with a Map Select Response message, and CV will attach to the target SgNB. When the buffered packets on the target SgNB are delivered to CV, the MeNB stops the U-plane transmission. After the completion of the handover, the relevant mapping information will be updated on the MEC server, the UE context information on the former SgNB will be released, and the route will be optimized from CV to TN. Figure 3 gives the complete process of intra-MeNB handover signaling.

It is obvious that the handover latency of intra-MeNB handover is possible to be zero. Since the MeNB plays the role of backup data plane transmission in the process of SgNB switching, a seamless handover can be ensured.

4.4. Inter-MeNB Handover Procedure. When a CV moves towards to a new MeNB control region (a new LSD) and satisfies a handover trigger condition, it will send a Map Select Request message to the source MeNB with its GAID. The source MeNB then sends a Map Update message through the overlay network to the target MeNB, which contains the CV's relevant $GAID_{TN}$ list and its RID so that the target MeNB will update its $GAID_{TN}$-to-RID_{MeNB} cache. Then, the target MeNB configures a new LAID for the CV and sends a Map Update Response message to the source MeNB. The source MeNB and the target MeNB send MAP Forwarding Request message to the source SgNB and the target SgNB, respectively. After that, the source SgNB sets up a data forwarding path with the target SgNB through sending a Map Forwarding Request message. If the target SgNB promises the access admission, the target SgNB will response Map Forwarding Response message to the source SgNB and the target MeNB. Meanwhile, the source SgNB sends a Map Forwarding Response message to the source MeNB, and then the previous transmitting data is forwarded to the target SgNB. At the moment, the source MeNB sends a Map Select Response message to the CV, and the CV will establish a connection with the target MeNB and then attach to the target SgNB. The buffering packets on the target SgNB will be forwarded to the CV once it is attached to itself. After the

Table 2: Context information.

CV	CV context set	$SgNB_1$	...	$SgNB_I$
v_1	$C^t_{v_1}$	$rsrp_{1,1}$	...	$rsrp_{1,I}$
v_2	$C^t_{v_2}$	$rsrp_{2,1}$	...	$rsrp_{2,I}$
...	...	...	...	...
v_K	$C^t_{v_k}$	$rsrp_{K,1}$	...	$rsrp_{K,I}$

completion of the handover, the relevant mapping information will be updated on the MEC server in the new LSD, and the $GAID_{CV}$-to-RID_{MeNB} mapping cache entries in the GW will also be updated. The UE context information on the former SgNB and MeNB will be released, and the route will be optimized from CV to TN. Figure 4 gives the complete process of inter-MeNB handover signaling.

Notice that the above handover procedures assume the existence of available SgNBs. If at a given time the coverage holes of SgNB exists, the U-plane transmission on the MeNB will immediately start. The original $GAID_{CV}$-to-RID_{SgNB} mapping entry will be deleted, and the CV will discard its LAID. Once an available SgNB meets the CV's connection requirements according to the algorithm described in the next section, the CV will reconfigure its LAID and performs a U-plane switch to the target SgNB. Figure 5 shows the signaling interactions of the fast U-plane switch procedures.

5. Handover Decision Algorithm

To further improve the handover performance, we propose a QoS-based network selection method to select the most suitable SgNB for the CV in this section. The algorithm jointly considers network balance and the network requirements of CV, which mainly includes two parts: the SgNB QoS-boundary conversion and the LSTM-based sojourn time prediction. Moreover, we redefine a dynamic handover trigger condition of the intra-MeNB handover, so as to improve the robustness of the intrahandover under the dense network scenario. The proposed handover selection algorithm will be executed on the MEC server based on the handover context information.

5.1. SgNB QoS-Boundary Conversion. We define a QoS circular coverage area centered at the serving/candidate SgNB. As for a specific CV, the SgNB can provide a satisfactory data transmission rate within its QoS circular area. We call the boundary of the circle QoS boundary. The derivation process of the SgNB QoS boundary of a CV jointly considers the real-time load of the SgNB and the network requirement of the served CV, which is shown as follows:

Suppose that each SgNB n_i has K UEs attached to it, the data requested rates of these UEs are $\{D_1, D_2, \cdots, D_K\}$. The network load of n_i can be precisely defined as

$$L_{n_i} = \sum_{n=1}^{K} \frac{1}{D_n}. \tag{1}$$

The effective maximum throughput of n_i is

$$Tp_{n_i} = \frac{1}{L_{n_i} + \left(1/D^{\max}_{n_i}\right)}, \tag{2}$$

where $D^{\max}_{n_i}$ is the maximum data transmit rate that n_i can provide at the moment. $D^{\max}_{n_i}$ can be predicted based on the relative measurement report parameters (RSRP and RSRQ) [39]. When a CV v_k requests for a data rate $D^{\text{req}}_{v_k}$, a n_i will be added to the candidate network set F if the condition $D^{\text{req}}_{v_k} < Tp_{n_i}$ is satisfied.

When a CV is in the coverage of a specific SgNB n_i, its received power is expressed as

$$P^{\text{RX}}_{ki} = P^{\text{TX}}_{ki} - P^{\text{LOSS}}_{ki} - FM + G, \tag{3}$$

where FM and P^{LOSS}_{ki} represent the fading margin and the path loss, respectively. G represents the antenna gain between n_i and v_k. According to the log-distance path loss model under the urban environment, P^{LOSS}_{ki} can be calculated as

$$P^{LOSS}_{ki} = \lambda log_{10}(r) + \beta log_{10}(f_c) + \gamma, \tag{4}$$

where λ, β, and γ are related to the surrounding road environment, f_c is the carrier frequency of the SgNB, and r represents the distance between n_i and v_k.

We use the average signal power P^{Ravg}_{ki} received by v_k over a period of time to represent P^{RX}_{ki}, which makes the QoS boundary of n_i more representative. P^{Ravg}_{ki} can be deduced based on the following Shannon's theorem:

$$D^{\max}_{n_i} = W\log_2\left(1 + \frac{\text{P}^{\text{Ravg}}_{\text{ki}}}{N}\right) \Leftrightarrow P^{\text{Ravg}}_{ki} = N\left(2^{D^{\max}_{n_i}/W} - 1\right), \tag{5}$$

where W and N represent the channel bandwidth and the noise power, respectively. We use the radius $r^{\text{QoS}}_{n_i}$ of the QoS circular coverage area to replace the parameter r in equation (4), and the value of $r^{\text{QoS}}_{n_i}$ can be obtained by combining (3)–(5):

$$r^{\text{QoS}}_{n_i} = 10^{\left(P^{\text{TX}}_{ki} - \beta\log_{10}(f_c) - \gamma - FM + G - P^{\text{Ravg}}_{ki}\right)/\lambda}. \tag{6}$$

5.2. LSTM-Based Vehicular Sojourn Time Prediction. The historical motion of a CV can be utilized to predict its future driving trends in future period to time. We can apply the LSTM neural network to learn features among CV's historical trajectories and predict the CV's future trajectory [40]. Based on the predicted trajectory, we further obtain the sojourn time of the CV within the coverage of each candidate network. We summarize the architecture of a normal LSTM cell and the calculation of each parameter in Figure 6.

In Figure 6, f_t, i_t, and o_t represent the forget, input, and output gates, respectively. The function of each gate are described in [40]. b_f, b_i, b_o, b_c are the corresponding variable

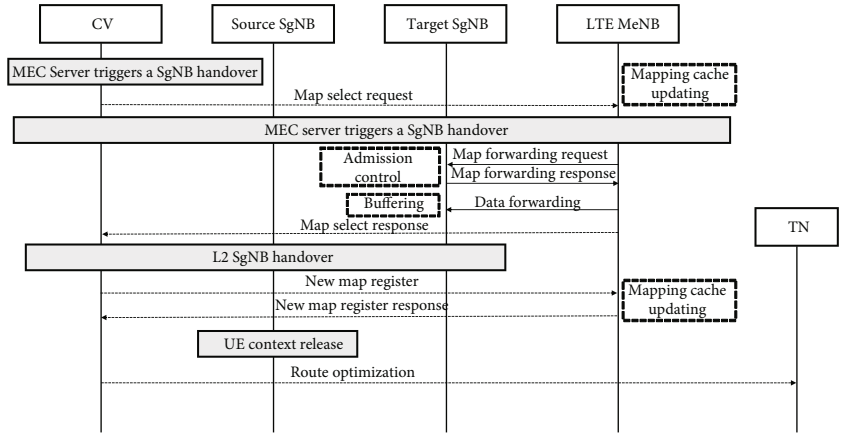


Figure 3: Intra-MeNB SgNB handover.

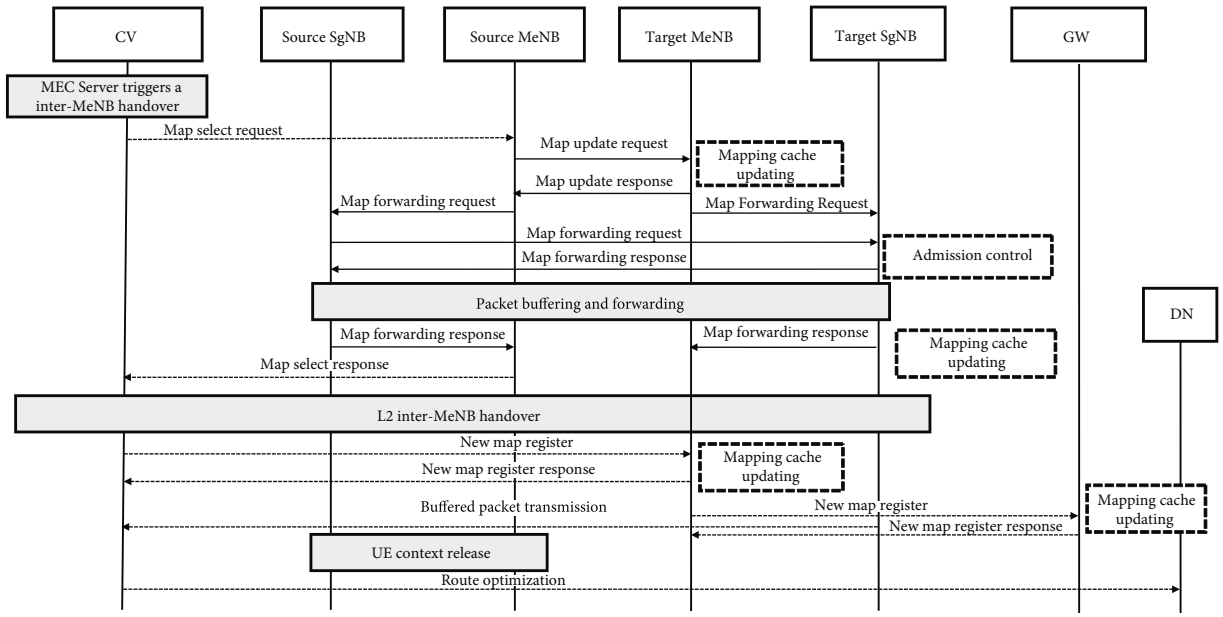


Figure 4: Inter-MeNB handover.

biases of $[h_{t-1}, x_t]$ in f_t, i_t, o_t, and $\tilde{C}_t$. W_f, W_i, W_o, W_c are the corresponding weights of $[h_{t-1}, x_t]$ in f_t, i_t, o_t, and $\tilde{C}_t$. Figure 7 illustrates the architecture of the proposed trajectory prediction model. The input of the prediction architecture is composed of 5 dimensions, i.e., the longitude and latitude coordinates $(x_{v_k}^t, y_{v_k}^t)$, the driving angle $\alpha_{v_k}^t$, the velocity $v_{v_k}^t$, and the acceleration $a_{v_k}^t$ of the vehicle. The final input set is represented as a vector $X_{v_k}(n)$ given by

$$X_{v_k}(n) = \left\{ x_{v_k}^t, y_{v_k}^t, \alpha_{v_k}^t, v_{v_k}^t, a_{v_k}^t \right\}, n = t - N + 1, \cdots, t, \quad (7)$$

where N denotes the length of the historical input time sequence. Note that the actual position sequence of the vehicle is transformed to Frenet coordinates in our prediction model for improving the adaptability of the training data and the accuracy of the prediction.

In this architecture, a fully connected (FC) layer made up of 256 cells can transforms the input data into 256 dimensional equal to the LSTM cell dimension of the following LSTM stack. Each dimension has a strong relationship with the input data. Specifically, according to the feedback network update parameters, we can determine the input dimensions which are more relevant to the predicted trajectory trend after the input dimension conversion.

The following LSTM stack consists of two LSTM layers each of which has 256 LSTM cells. The output vector from the first LSTM layer is an input of the second LSTM layer. The function of the LSTM stack is to extract higher-level features of the input time series. The output of the LSTM stack is combined by the FC stack with two FC layers in order to reduce the data dimension. Meanwhile, the input sequences are also fed to a 64-dimensional FC layer, bypassing the above network connections. The outputs of this FC layer and the previous FC stack are directly fed to the output stack to obtain the final predicted future states. This kind of design

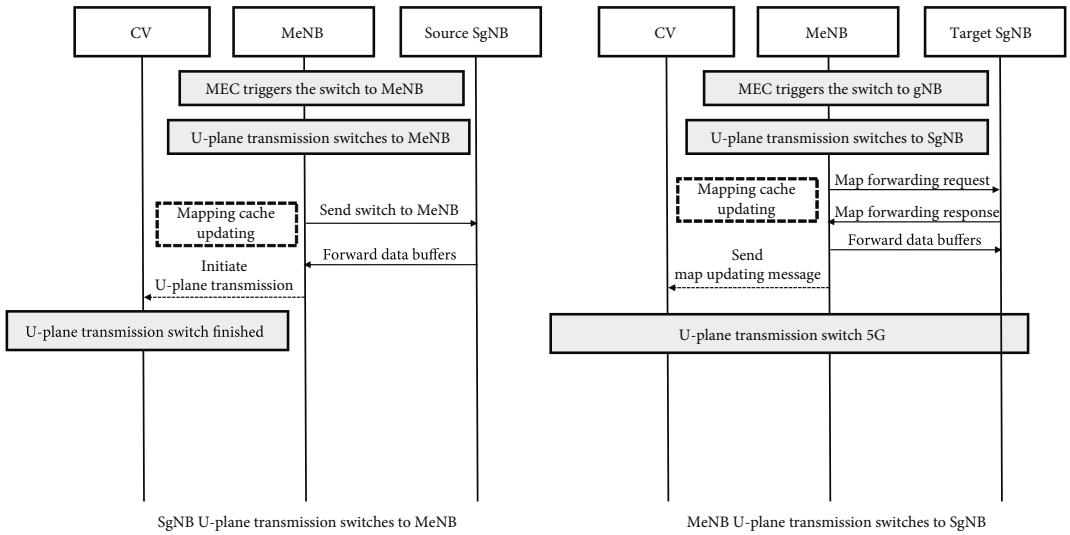


Figure 5: U-plane fast switch procedure.

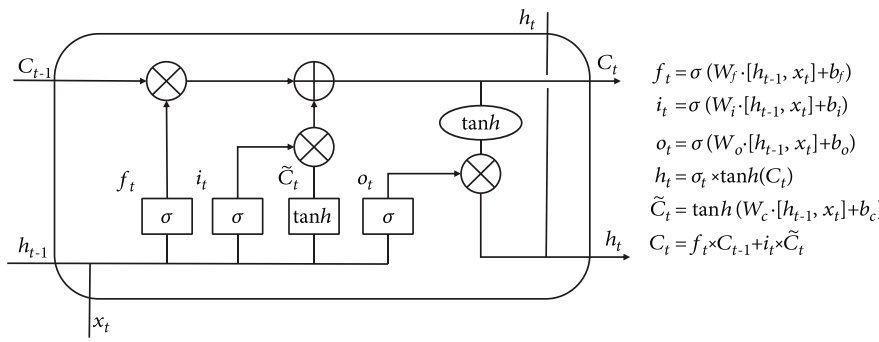


Figure 6: LSTM cell architecture.

can establish a closer correlation between the current input states and the prediction output sequences, which boosts the training speed and improves the prediction accuracy of the model.

The output $Y_{v_k}(n)$ of the vehicular trajectory prediction model is shown as follows:

$$Y_{v_k}(n) = \left\{x_{v_k}^t, y_{v_k}^t\right\}, n = t+1, \cdots, t+M. \tag{8}$$

Finally, the Frenet coordinates of the predicted trajectory will be converted back to the actual coordinates.

Based on the QoS boundary coverage of each candidate SgNB and the LSTM-based vehicular sojourn time prediction model, we can obtain the sojourn time $t_{n_i}^{v_k}$ of a CV v_k in the QoS boundary coverage of the candidate SgNB n_i. For the network access entities of our proposed network architecture in Figure 8, green dotted circles represent the QoS coverage range of SgNBs.

Assuming ABCD is a predicted future driving track of v_k , point B and point D are two predicted positions within the QoS coverage of SgNB3 that are closest to the QoS boundary. Thus, the sojourn time $t_{n_3}^{v_k}$ of the v_k in SgNB3 can be determined as $t_D^{v_k} - t_B^{v_k}$, where $t_B^{v_k}$ and $t_D^{v_k}$ are the time when the v_k reaches the positions B and D.

5.3. Dynamic Time Threshold Condition. Since the heterogeneity of 5G gNBs may lead to highly dynamic handover context, we now redefine the trigger condition of the intra-MeNB handover based on the SgNB RSRP measurement reports, the predefined time threshold (TTT), and the QoS coverage area sojourn time $t_{n_i}^{v_k}$.

A SeNB in the candidate set F can build a link with the CV under each of the following two conditions that the SeNB has a better RSRP than the serving one, and it has also a higher sojourn time than the predefined threshold T_{th}. If both conditions hold, the MEC server checks for Δ seconds and then triggers the intra MeNB handover. Notice that the SgNB with the highest $t_{n_i}^{v_k}$ in the set F can be chosen as the original target SgNB. During the Δ, if the RSRP of the serving SgNB becomes the highest or $t_{n_i}^{v_k} < T_{th}$, the handover will be cancelled. The Δ will be reset based on equation (9) when a new handover condition holds. rsrp_c is the RSRP value of the serving SgNB. $\text{rsrp}_{\max}$ and $\text{rsrp}_{\min}$ represent the maximum and minimum RSRP values in the candidate network set F, respectively. $\Delta_{\max}$ and $\Delta_{\min}$ are static values based on historical experience. Then, we have

$$\Delta = \Delta_{\max} - \frac{\text{rsrp}_c - \text{rsrp}_{\min}}{\text{rsrp}_{\max} - \text{rsrp}_{\min}} (\Delta_{\max} - \Delta_{\min}). \tag{9}$$

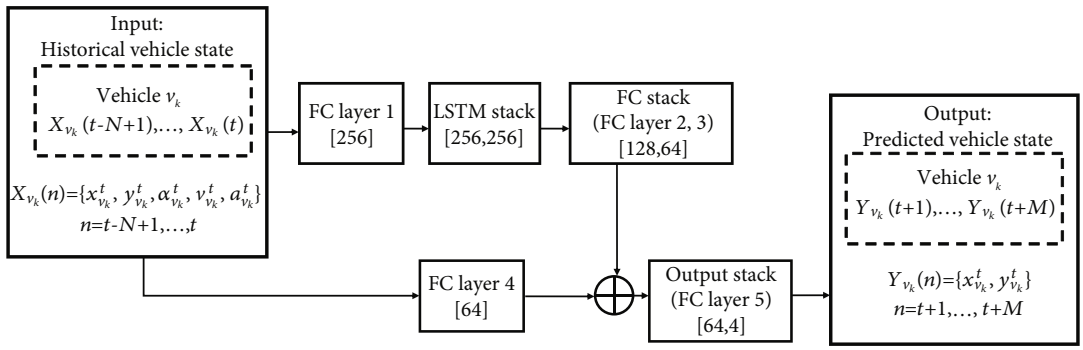


Figure 7: The architecture of the LSTM-based trajectory prediction model.

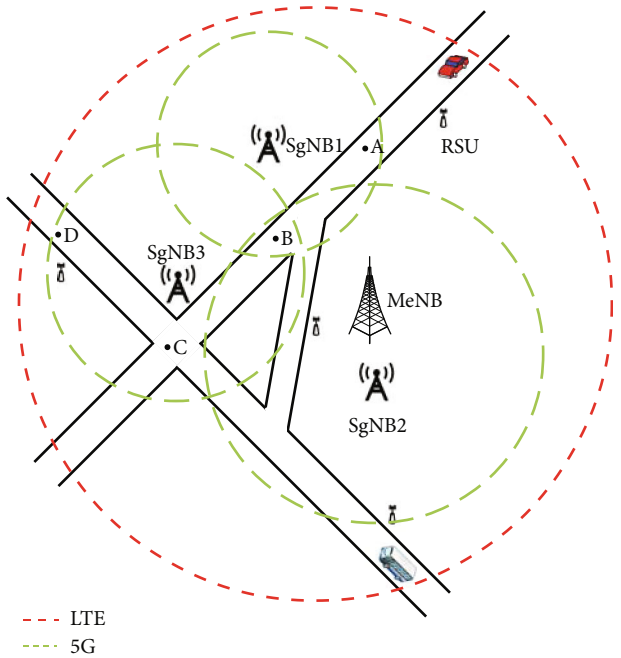


Figure 8: Network access entities of our proposed network architecture.

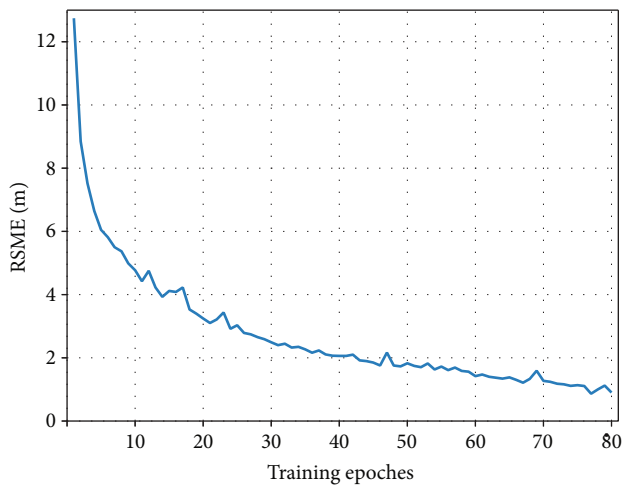


Figure 9: RSME versus training epoches of the proposed model.

The Δ value will become smaller when the difference in RSRP between the serving SgNB and the target SgNB becomes larger. It is also worth noting that, when a CV detects a radio link failure (RLF) [41] with the serving SgNB and there happens to be no suitable candidate SgNB in the set F, the terminal maintains the C/U-plane transmission with the MeNB. Therefore, the impact of an SgNB on RLF is moderate.

5.4. Algorithm Complexity. We analyze the complexity of our proposed handover decision algorithm in two parts.

Firstly, we deduce the complexity of training the LSTM-based vehicular sojourn prediction model. As for an LSTM layer, its training complexity is based on its input dimension I and output dimension U [42], which can be presented as $\mathcal{O}(4(\mathrm{IU} + U^2 + U))$. Meanwhile, the training complexity of the fully connected layer is $\mathcal{O}(\mathrm{IU})$, so that the training complexity of our prediction model can be presented as $\mathcal{O}(q_I q_1 + 8(q_1 q_L + q_L^2 + q_L) + \sum_{m=2}^{4} q_m q_{m+1} + q_5 q_O)$, where q_m $(m \in \{1, 2, 3, 4, 5\})$ is the cell number of each fully connected layer in our model, respectively. q_L is the cell number of each LSTM layer. q_I and q_O are the numbers of input and output cells. Considering that the prediction model has already completed its training process before being applied, it will not bring extra computing cost during the handover decision process.

Moreover, we analyze the complexity of the decision process. Considering that our algorithm selects the gNB with the largest sojourn time from the candidate network set F, the complexity of each check can be easily deduced as $\mathcal{O}(|F|\log_2(|F|))$, where $|F|$ is the cardinality of set F.

6. Simulation Results

This section first validates our proposed model and then conducts the performance evaluation study.

6.1. Model Validation. In this paper, we use the NGSIM data set [43] to collect relevant vehicle trajectory in US101 sections for the training and testing of our proposed trajectory prediction model. We first randomly select 70% of data (i.e., 4269 trajectories) for training, 20% of data for validation (i.e., 1220 trajectories), and the remaining 10% of data (i.e., 610 trajectories) for testing. The sampling frequency of the dataset is 1 Hz. The model training is executed on GPU

Table 3: RMSE values of the proposed model.

	Prediction horizon									
	1 s	2 s	3 s	4 s	5 s	6 s	7 s	8 s	9 s	10s
RSME in latitude	0.041	0.072	0.079	0.084	0.088	0.096	0.11	0.12	0.17	0.31
RSME in longitude	0.65	0.86	0.99	1.10	1.15	1.25	1.38	1.66	2.14	4.02
Total RSME	0.66	0.86	1.00	1.11	1.15	1.26	1.38	1.67	2.15	4.03

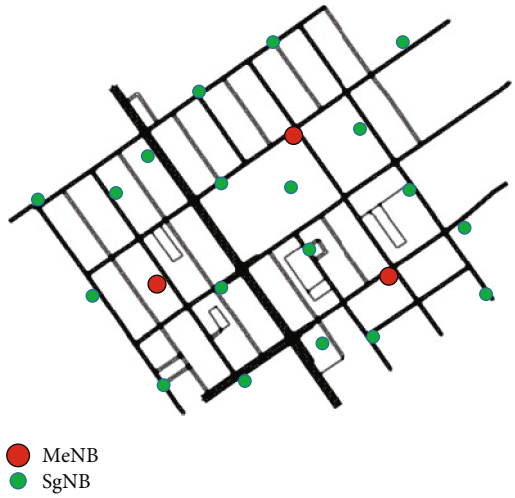


Figure 10: Real urban road scenario abstracted by SUMO.

using the Pytorch with a batch size of 128. We have trained the model for 4 epochs, and the whole dataset has been processed for 20 times, which results in 80 effective epochs. The prediction accuracy of the model is measured by the root mean square error (RMSE) between the predicted value and the actual expected value of the position. The variation of RSME loss with training epoches is shown in Figure 9, and the RSME of our model in different prediction horizons is provided in Table 3.

When a CV is traveling at an average velocity of 30 km/h, the average predicted sojourn time error is about 0.486 s in a 10 s prediction horizon. Since the actual sojourn time of the CV is about 24 s in a SgNB with coverage radius of 150 m, the prediction error of the sojourn time is negligible.

6.2. Handover Performance Evaluation. We build the network simulation environment on the NS-3 simulator [44]. A real road traffic environment around the US101 section is abstracted by SUMO [45] as shown in Figure 10. LTE MeNBs and 5G SgNBs with different configuration parameters are deployed in this area. The SgNBs are wired to their MeNB which act as an LMA. The MeNBs are wired to the GW. The CVs' trajectories are randomly generated by SUMO based on the abstracted road traffic environment, and we set the value of the velocity as multiple constant values. The values of the main simulation parameters are shown in Table 4. We now define some fundamental performance metrics as follows:

(i) Handover latency: it is defined as the average time duration from the time when a CV and the network system needs to update the relevant mapping information to the time when the CV receives the packet from the target access network entity

(ii) Handover times: it is defined as the average time over handover events happening for a certain simulation time

(iii) Packet loss rate: It can be expressed as $X_{\text{tot}} - X_{\text{rec}}/X_{\text{tot}}$, where X_{tot} and X_{rec} are the number of packets sent by the TNs and that received by the CVs during a session, respectively

(iv) Mobility management load: it is the average number of mobility management packets processed at the relevant control entity per minute under the scenario of CVs' mobility

(v) gNB utilization rate: it is the average ratio of the time that the CV is connected to the serving SgNB (T_{used}) to the time that the SgNB is available for CV to attach (T_{usable}), which can be expressed as $T_{\text{used}}/T_{\text{usable}}$

To illustrate the efficiency of our proposed mechanism, we conduct comparison study with the ILS mobility management mechanism (LISP-MN) [26] using the A2A4 [38] handover algorithm.

6.2.1. Handover Latency. As shown in Figure 11, we explore the average handover latency under two types of mechanisms with $v_{v_k} = 40$ km/h. Since the vehicular velocity is usually required to be no more than 40 km/h in the urban traffic scenario, the vehicular velocity 40 km/h is a typical example here. It can be observed from Figure 11 that our proposed mechanism has the lowest handover latency in comparison with the LISP-MN DC and LISP-MN Hard handover. This can be explained as follows. Firstly, the decentralized mapping system of our proposed mechanism significantly reduces the mapping update latency caused by the backbone transmission of the control signaling. Meanwhile, the DC technology provides the backup transmission of MeNB, which can reduce the link interruption time caused by SgNB handover. This can also explains the reason why the handover latency of LISP-MN DC is lower than that of LISP-MN Hard handover as shown in Figure 11.

Our proposed mechanism reduces the intra-MeNB handover latency up to 57.1 percent and the inter-MeNB handover latency up to 61.9 percent compared with the LISP-MN mechanism.

Table 4: Simulation parameters.

Parameters	Values
Number of base stations	22 (3 MeNB, 19 SgNB)
Number of GWs	1
Transmitting power of base stations	23 ~ 46 dBm
Maximum coverage radius of MeNBs	500 m
Maximum coverage radius of SgNBs	150 m
Operating frequency of base stations	MeNB: 1.85 GHz, SgNB: 700 MHz
CV velocity (v_{v_k})	{30, 40, 50} km/h
Coefficient of the path loss model	$\lambda = 16.7, \beta = 18.2, \gamma = 38.77$
Fading margin (FM)	8 dB
Antenna gain (G)	13 dB
$\Delta_{\max}$	120 ms
$\Delta_{\min}$	15 ms
Transfer protocol	UDP
MTU	1300 byte
CIT updating cycle	20 ms
The cost of each mobility management on the GW	24
SgNB to MeNB link	Delay: 5 ms Link type: point to point
MeNB to GW link	Delay: 10 ms Link type: point to point
Intra-SgNB to SgNB link	Delay: 5 ms Link type: point to point
Simulation time	300 s
Simulation area	1000 m*900 m

6.2.2. Handover Times. As shown in Figure 12, we examine how total network handover times vary with v_{v_k} during the simulation process. We can see from Figure 12 that for each fixed v_{v_k}, total network handover times under our proposed mechanism are lower than the times under the traditional handover algorithm (A2A4). This is because the A2A4 handover algorithm only adopts the signal strength as the handover decision condition, while the CV mobility characteristic can result in unnecessary handovers, which largely affects the future network access. Our proposed mechanism not only guarantees the link quality but also consider the effect of cell sojourn time on the handover times. The SgNB with the maximum sojourn time is selected as the target access entity to minimize the possibility of unnecessary handover. We can also observe from Figure 12 that a larger v_{v_k} leads to higher handover times due to the fact that the higher velocity leads to a shorter cell sojourn time and thus higher frequency of handovers. However, our proposed mechanism can curb the increase of the handover frequency compared with the traditional mechanism.

6.2.3. Packet Loss Rate. As shown in Figure 13, we investigate how the packet loss rate varies with different packet arrival rates λ. We can see from the Figure 13 that for each fixed v_{v_k}, the packet loss rate under our proposed mechanism is lower than that under the traditional LISP-MN mechanism. This is due to the following reason. The decentralized mapping system in the proposed mechanism brings relative low update latency, which reduces the session recovery time and relieves the packet loss during the handover procedure. Meanwhile, the addition of backup data forwarding and fast handover signaling methods in the proposed mechanism further improves the performance of the packet loss rate.

Due to the relevant optimization, our proposed mechanism reduces the packet loss rate up to 51 percent in an extreme situation ($\lambda = 100$packets/s, $v_{v_k} = 50$ km/h) compared with the LISP-MN mechanism.

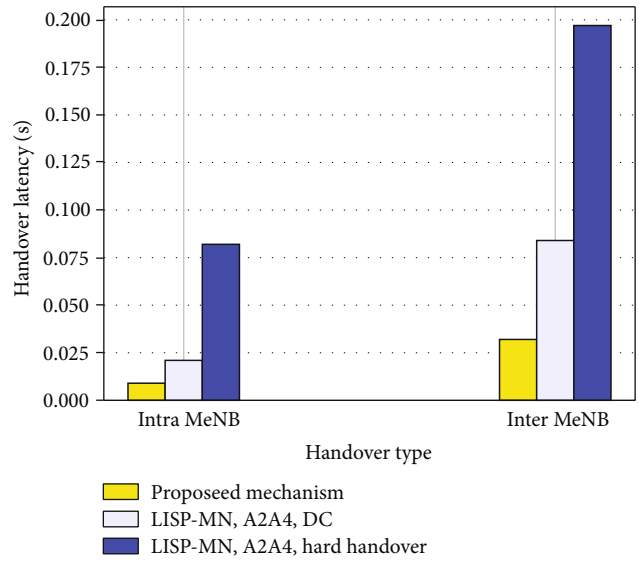


Figure 11: Handover latency under different handover types.

6.2.4. Mobility Management Load. As shown in Figure 14, we explore how the total mobility management load on the GW varies with v_{v_k}. We can see from Figure 14 that for each fixed v_{v_k}, the total mobility management load on the GW under our proposed mechanism is lower than that under the traditional LISP-MN mechanism. It is because under the decentralized mapping update strategy, the intra-MeNB mapping update overhead has been offloaded to the local MEC server. We can also observe from Figure 14 that our proposed mechanism can alleviate the load caused by the velocity and the number of CVs. Specifically, the total mobility management load on the GW can reduced up to 83.5 percent in a high mobility and mass CV scenario with the setting of $v_{v_k} = 50$ km/h and CV number = 200 compared with the LISP-MN mechanism.

6.2.5. SgNB Utilization Rate. Finally, we explore how the average SgNB utilization rate varies with data requested rate as shown in Figure 15. It can be seen from Figure 15 that for each fixed v_{v_k}, the average SgNB utilization rate under our proposed mechanism is higher than that under the traditional LISP-MN mechanism. This is because the proposed

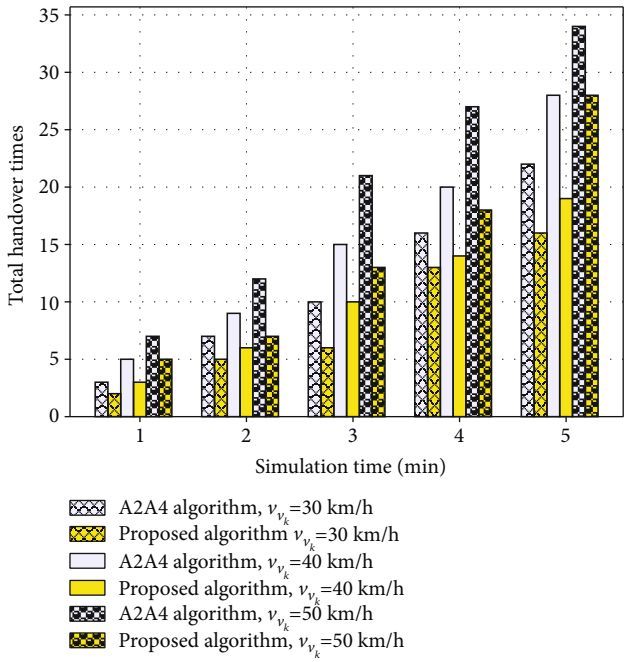


Figure 12: Total handover times under different v_{v_k}.

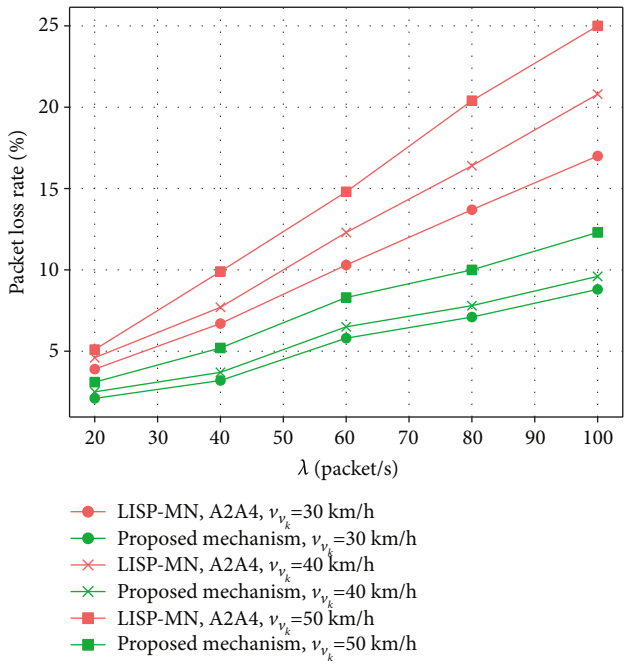


Figure 13: Packet loss rate versus λ under different v_{v_k}.

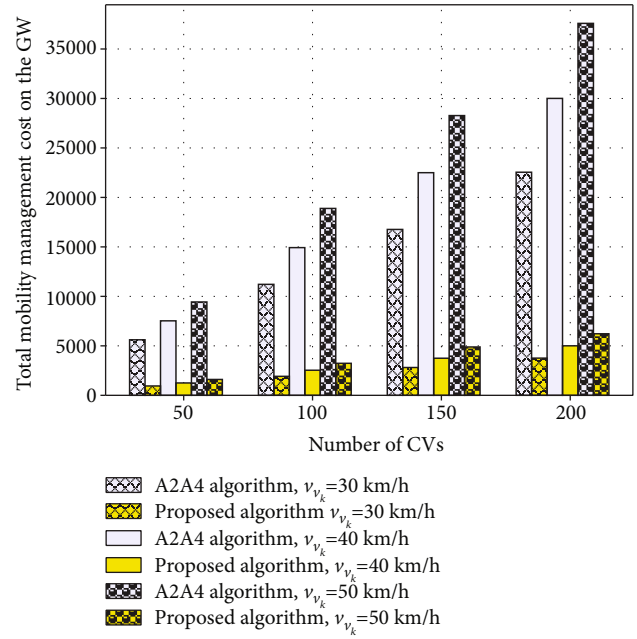


Figure 14: Total mobility management load on the GW versus v_{v_k}.

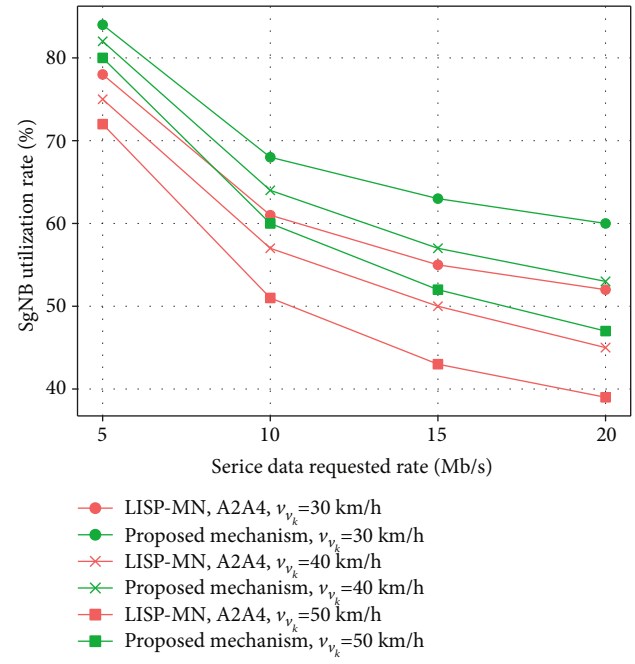


Figure 15: Average SgNB utilization rate versus data request rate.

mechanism selects the SgNB with the longest sojourn time as the target SgNB and ignores the SgNBs which do not meet the load requirements. A further observation from Figure 15 indicates that both the increases of data requested rate and velocity will reduce the SgNB utilization. We know that the CV will gain the access admission to the target SgNB when it has sufficient network resources. The high requested rate can increase the occupation of the network resource and cause the access reject during the handover process, which leads to a decreased T_{used}. Meanwhile, the increase of velocity can also reduce the cell sojourn time of the CV, which also results in a decrease of T_{used}.

7. Conclusion

This paper proposed an optimal decentralized mobility management mechanism for the dense 5G networks. Under this mechanism, we first designed an LMA-based handover

management architecture, which jointly applies the technical advantages of ILS, dual connectivity, and MEC to realize a low signaling cost mobility management under the dense gNB scenarios. Then, we proposed a QoS-based handover decision algorithm to ensure network balance and improve the network utilization, which unitizes a predefined QoS boundary conversion method involving an LSTM-based vehicular sojourn time prediction model. Moreover, we redefined the dynamic trigger condition in the handover algorithm to enhance the robustness of the intra-MeNB handover decision in highly different link scenarios with the heterogeneity of SgNBs. Simulation results illustrate that the LSTM-based prediction model in our proposed handover decision algorithm can achieve a low trajectory prediction error. Meanwhile, our proposed mobility management mechanism can significantly reduce the handover latency, the handover times, the packet loss rate, and the mobility management load and also improve the gNB utilization rate compared to a classic traditional mechanism.

Data Availability

All data, models, and code generated or used during the study appear in the submitted article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This paper is supported by the National Key Research and Development Project of China (No. 2019YFF0303101).

References

[1] B. Feng, H. Zhang, H. Zhou, and S. Yu, "Locator/identifier split networking: a promising future internet architecture," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, pp. 2927–2948, 2017.

[2] S. Paul, R. Jain, and J. Pan, "An identifier/locator split architecture for exploring path diversity through site multi-homing - a hybrid host-network cooperative approach," in *2010 IEEE International Conference on Communications*, pp. 1–5, Cape Town, South Africa, 2010.

[3] C. R. Storck and F. Duarte-Figueiredo, "A survey of 5G technology evolution, standards, and infrastructure associated with vehicle-to-everything communications by Internet of Vehicles," *IEEE Access*, vol. 8, pp. 117593–117614, 2020.

[4] G. Pujolle, *Software Networks: Virtualization, SDN, 5G and Security*, Wiley, Hoboken, NJ, USA, 2015.

[5] X. Ge, S. Tu, G. Mao, C. Wang, and T. Han, "5G ultra-dense cellular networks," *IEEE Wireless Communications*, vol. 23, no. 1, pp. 72–79, 2016.

[6] P. Dong, T. Zheng, S. Yu, H. Zhang, and X. Yan, "Enhancing vehicular communication using 5G-enabled smart collaborative networking," *IEEE Wireless Communications*, vol. 24, no. 6, pp. 72–79, 2017.

[7] S. Chen, F. Qin, B. Hu, X. Li, and Z. Chen, "User-centric ultra-dense networks for 5G: challenges, methodologies, and directions," *IEEE Wireless Communications*, vol. 23, no. 2, pp. 78–85, 2016.

[8] C. White, D. Lewis, D. Meyer, and D. Farinacci, "Lisp mobile node," *IETF Internet draft*, 2011, http://tools.ietf.org/html/draft-meyer-lisp-mn-06.

[9] R. Moskowitz, T. Heer, P. Jokela, and T. Henderson, *Host Identity Protocol Version 2 (HIPv2)*, IETF, Fremont, CA, 2015, Tech. Rep. RFC 7401.

[10] J. Pan, S. Paul, R. Jain, and M. Bowman, "MILSA: a mobility and multihoming supporting identifier locator split architecture for naming in the next generation Internet," in *IEEE GLOBECOM 2008-2008 IEEE Global Telecommunications Conference*, pp. 1–6, New Orleans, LA, USA, 2008.

[11] B. Yang, X. Chen, J. Xie, S. Li, Y. Zhang, and J. Yang, "Multicast design for the mobility first future Internet architecture," in *2019 International Conference on Computing, Networking and Communications (ICNC)*, pp. 88–93, Honolulu, HI, USA, 2019.

[12] H. Zhang, W. Quan, H. Chao, and C. Qiao, "Smart identifier network: a collaborative architecture for the future internet," *IEEE Network*, vol. 30, no. 3, pp. 46–51, 2016.

[13] V. P. Kafle, H. Otsuki, and M. Inoue, "An ID/locator split architecture for future networks," *IEEE Communications Magazine*, vol. 48, no. 2, pp. 138–144, 2010.

[14] F. Qiu, X. Li, and H. Zhang, "Mobility management in identifier/locator split networks," *Wireless Personal Communications*, vol. 65, no. 3, pp. 489–514, 2012.

[15] H. Luo, H. Zhang, and C. Qiao, "Efficient Mobility Support by Indirect Mapping in Networks With Locator/Identifier Separation," *IEEE Transactions on Vehicular Technology*, vol. 60, no. 5, pp. 2265–2279, 2011.

[16] H. Zhang, H. Luo, and H. Chao, "Dealing with mobility-caused outdated mappings in networks with identifier/locator separation," *IEEE Transactions on Emerging Topics in Computing*, vol. 4, no. 2, pp. 199–213, 2016.

[17] H. Zhang, P. Dong, W. Quan, and B. Hu, "Promoting efficient communications for high-speed railway using smart collaborative networking," *IEEE Wireless Communications*, vol. 22, no. 6, pp. 92–97, 2015.

[18] D. Jiang, L. Huo, Z. Lv, H. Song, and W. Qin, "A joint multi-criteria utility-based network selection approach for vehicle-to-infrastructure networking," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 10, pp. 3305–3319, 2018.

[19] M. Gohar and S.-J. Koh, "A distributed mobility control scheme in LISP networks," *Wireless Networks*, vol. 20, no. 2, pp. 245–259, 2014.

[20] M. Gohar and S. J. Koh, "Network-based distributed mobility control in localized mobile LISP networks," *IEEE Communications Letters*, vol. 16, no. 1, pp. 104–107, 2012.

[21] F. Giust, L. Cominardi, and C. J. Bernardos, "Distributed mobility management for future 5G networks: overview and analysis of existing approaches," *IEEE Communications Magazine*, vol. 53, no. 1, pp. 142–149, 2015.

[22] T. Guo, A. u. Quddus, N. Wang, and R. Tafazolli, "Local Mobility Management for Networked Femtocells Based on X2 Traffic Forwarding," *IEEE Transactions on Vehicular Technology*, vol. 62, no. 1, pp. 326–340, 2013.

[23] D. Pacheco-Paramo, I. F. Akyildiz, and V. Casares-Giner, "Local anchor based location management schemes for small cells in HetNets," *IEEE Transactions on Mobile Computing*, vol. 15, no. 4, pp. 883–894, 2016.

[24] L. Yi, H. Zhou, D. Huang, and H. Zhang, "An analytical study of distributed mobility management schemes with a flow duration based model," *Journal of Network & Computer Applications*, vol. 41, pp. 351–357, 2014.

[25] X. Lin, R. K. Ganti, P. J. Fleming, and J. G. Andrews, "Towards understanding the fundamentals of mobility in cellular networks," *IEEE Transactions on Wireless Communications*, vol. 12, no. 4, pp. 1686–1698, 2013.

[26] S. Barbera, P. H. Michaelsen, M. Säily, and K. Pedersen, "Improved mobility performance in LTE co-channel hetnets through speed differentiated enhancements," *2012 IEEE Globecom Workshops*, pp. 426–430, 2012.

[27] W. Qi, Q. Song, S. Wang, Z. Liu, and L. Guo, "Social prediction-based handover in collaborative-edge-computing-enabled vehicular networks," *IEEE Transactions on Computational Social Systems*, vol. 9, no. 1, pp. 207–217, 2022.

[28] K. Kitagawa, T. Komine, T. Yamamoto, and S. Konishi, "Performance evaluation of handover in LTE-advanced systems with pico Cell Range Expansion," in *2012 IEEE 23rd International Symposium on Personal, Indoor and Mobile Radio Communications - (PIMRC)*, pp. 1071–1076, Sydney, NSW, Australia, 2012.

[29] G. Liu, Y. Huang, Z. Chen, L. Liu, Q. Wang, and N. Li, "5G deployment: standalone vs. non-standalone from the operator perspective," *IEEE Communications Magazine*, vol. 58, no. 11, pp. 83–89, 2020.

[30] 3GPP, "Evolved universal terrestrial radio access (E-UTRA) and evolved universal terrestrial radio access network (E-UTRAN); overall description, stage 2," 2016.

[31] P. Hsieh, W. Lin, K. Lin, and H. Wei, "Dual-Connectivity Prevenient Handover Scheme in Control/User-Plane Split Networks," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 4, pp. 3545–3560, 2018.

[32] T. Taleb, K. Samdanis, B. Mada, H. Flinck, S. Dutta, and D. Sabella, "On multi-access edge computing: a survey of the emerging 5G network edge cloud architecture and orchestration," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1657–1681, 2017.

[33] G. Hong, Q. Wen, and W. Su, "A modified vehicular handover scheme in non-standalone 5G networks with the assistance of multi-access edge computing," in *2021 International Conference on Networking and Network Applications (NaNA)*, pp. 174–180, Lijiang City, China, 2021.

[34] F. Giust, C. J. Bernardos, and A. de la Oliva, "Analytic evaluation and experimental validation of a network-based IPv6 distributed mobility management solution," *IEEE Transactions on Mobile Computing*, vol. 13, no. 11, pp. 2484–2497, 2014.

[35] T.-T. Nguyen and C. Bonnet, "DMMS: a flexible architecture for multicast listener support in a distributed mobility management environment," *Computer Networks*, vol. 94, pp. 129–144, 2016.

[36] D. Xenakis, N. Passas, L. Merakos, and C. Verikoukis, "Mobility management for femtocells in LTE-advanced: key aspects and survey of handover decision algorithms," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 1, pp. 64–91, 2014.

[37] European Telecommunications Standards Institute, *Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 2: Specification of Cooperative Awareness Basic Service*, EN 302 637-2 V1.3.2, European Telecommunications Standards Institute, 2014.

[38] European Telecommunications Standards Institute, *LTE; Evolved Universal Terrestrial Radio Access (E-UTRA); Radio Resource Control (RRC); Protocol Specification*, ETSI TS 136 331, European Telecommunications Standards Institute, 2017.

[39] R. Falkenberg, K. Heimann, and C. Wietfeld, "Discover your competition in LTE: client-based passive data rate prediction by machine learning," in *GLOBECOM 2017-2017 IEEE Global Communications Conference*, pp. 1–7, Singapore, 2017.

[40] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[41] 3GPP, "TS 37.340; Evolved universal terrestrial radio access (E-UTRA) and NR; multi-connectivity; stage 2 (Release 15)," 2017.

[42] T. Ergen and S. S. Kozat, "Online training of LSTM networks in distributed systems for variable length data sequences," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 10, pp. 5159–5165, 2018.

[43] US Department of Transportation, *Next generation simulation*, US Department of Transportation, Washington, DC, USA, 2008, http://www.ngsim.fhwa.dot.gov.

[44] Network Simulator version 3https://www.nsnam.org.

[45] Simulation of Urban Mobility (SUMO)http://sumo.sourceforge.net.

Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 6495546, 12 pages
https://doi.org/10.1155/2022/6495546

Research Article

Novel Shuffling Countermeasure for Advanced Encryption Standard (AES) against Profiled Attack in Mobile Multimedia Services

JongHyeok Lee,[1] **Jiyoon Kim**,[1] **and Dong-Guk Han**[1,2]

[1]*Department of Financial Information Security, Kookmin University, 02707 Seoul, Republic of Korea*
[2]*Department of Information Security, Cryptology and Mathematics, Kookmin University, 02707 Seoul, Republic of Korea*

Correspondence should be addressed to Dong-Guk Han; christa@kookmin.ac.kr

Received 12 May 2022; Accepted 21 June 2022; Published 13 July 2022

Academic Editor: Yuanlong Cao

Mobile multimedia services are gaining popularity among many users by developing wireless communication and mobile devices. Mobile multimedia has alleviated conventional multimedia's time and space limits, making it easier for consumers to access services and meet content demands. However, cyber risks lie in the shadows of the expansion of mobile multimedia services, threatening to continue wreaking havoc. Although various methods exist to defend against these cyber threats, side-channel analysis has remained a critical challenge in the current approaches that rely on cryptographic algorithms. Nowadays, research on deep learning-based side-channel analysis is receiving much attention. Attacks are constantly performed against implementations, to which existing countermeasures against traditional side-channel analysis are applied, using various artificial neural network structures. However, while studies on the implementations to which masking and simple hiding schemes using jitter are active, studies on the implementations to which the shuffling scheme or the random insertion of dummy operations scheme are applied have been relatively less attention. In a previous study, Lee and Han has used deep learning to distinguish between real and dummy operations in an implementation that combined shuffling scheme and random insertion of dummy operations scheme. They also proposed countermeasures against their attacks. However, they did not choose an appropriate environment that is as close to noise-free as possible, and their countermeasure still has flaws. Therefore, in this study, we analyze the causes of vulnerability of the previous countermeasure and propose a novel countermeasure that can completely solve them. The novel countermeasure is a method of uniformly applying shuffling schemes and random insertion of dummy operation schemes to byte-independent and byte-dependent operations of an advanced encryption standard, respectively. It was confirmed that our countermeasure is safe from attackers who perform profiled attacks even in an experimental environment with almost no noise.

1. Introduction

Mobile multimedia services have risen in response to the advancement of mobile communication technologies and increasing demand for content. Especially with the introduction of 5G, multimedia content can now be quickly delivered to users with high communication capacity, transmission speed, and low latency. Accordingly, the global mobile subscriber base is predicted to grow from 5.1 billion in 2018 to 5.7 billion in 2023, representing an increase from 66% of the worldwide population in 2018 to 71% in 2023 [1]. However, as the mobile multimedia industry grows, cyber threats against it become more diverse and complex. These threats can cause various damages to both customers and service providers. In this paper, we focus on side-channel analysis among the cyber threats of mobile multimedia services.

Side-channel analysis reveals secret information based on the fact that the power consumption of cryptographic devices depends on intermediate values of the cryptographic algorithms. These dependencies are of two types, data and operation dependency [2]. Typical attack methods utilizing

data dependency are differential and correlation power analysis [3, 4]. On the other hand, other methods use operation dependency is simple power analysis [3]. Simple power analysis recovers secret information using the difference in operation according to the secret information for asymmetric cryptographic algorithms or is mainly used to distinguish the operation of symmetric cryptographic algorithms to enable intensive side-channel trace collection. Therefore, simple power analysis uses a single trace, whereas differential and correlation power analyses use many traces because these are attacks that recover secret information using statistical techniques.

Countermeasures against side-channel analysis break data dependency or operation dependency of side-channel information. Software countermeasures usually break data dependency which is divided into masking [5, 6] and hiding schemes [2, 7]. Masking schemes overlay random values on an intermediate value to make it seem to be random. This makes it impossible for an attacker to infer the intermediate value. On the other hand, hiding schemes randomize the execution time of operations to prevent an attacker from estimating the operation time. Masking schemes logically block specific attacks, and for an attacker to attack a target that the masking schemes block, high-level attacks must be used to neutralize the masking schemes, compared with hiding schemes where it only increases an attack complexity by increasing the number of traces required by the attacker.

There are several methods to reduce the attack complexity increased by hiding schemes. One is the alignment method, which is commonly used. This type of method comprises static alignment, elastic alignment, and alignment schemes using pattern recognition or hidden Markov models [2, 8–10]. Recently, as research on deep learning-based side-channel analysis (DLSCA) progresses, studies have been published where some deep learning-based side-channel analyses neutralize hiding schemes [11–13]. In addition, studies on DLSCA against shuffling and dummy operations are being actively conducted [14–16]. However, they mainly deal with desynchronization due to jitter, and so on, rather than shuffling scheme or the random insertion of dummy operations scheme. Lee and Han performed machine learning-based side-channel analysis for the first time on a target in which the shuffling scheme and the random insertion of dummy operation schemes were used [17]. They showed that an attack was possible and at the same time suggested a countermeasure.

1.1. Our Contributions. In this study, we question the safety of the previous countermeasure. In a previous study, the authors experimentally demonstrated the safety of the proposed countermeasure but their experimental environment was noisy when compared to this present study [17]. Moreover, the previous countermeasure directly refers to the shuffled order array exposing the vulnerability. Because of the noise in the experimental environment, this vulnerability did not appear in the experimental results.

The current study shows that the previous countermeasure is insecure by performing profiled attacks on the ChipWhisperer-Lite board [18], which is considered an ideal environment with particle noise. Furthermore, this shows that the previous countermeasure is ineffective and reveals the cause of its weakness.

Input: IN[32], ORD[32]
Output: OUT[32]
1: for $i \longleftarrow$ 0 to 31do
2: OUT[ORD[i]]$\longleftarrow$Sbox[IN[ORD[i]]]
3: end for

Algorithm 1: Pseudocode for the previous countermeasure [19]

```
1; OUT[ORD[i]] = Sbox[IN[ORD[i]]]
2 movw   r28, r24
3 ldi    r26, 0x13
4 ldi    r27, 0x22
5 ldi    r20, 0x33
6 ldi    r21, 0x22
7 ld     r18, X+
8 ldi    r19, 0x00
9 movw   r30, r28
10 add    r30, r18
11 adc    r31, r19
12 ld     r30, Z
13 ldi    r31, 0x00
14 subi   r30, 0xF6
15 sbci   r31, 0xDF
16 ld     r25, Z
17 movw   r30, r22
18 add    r30, r18
19 adc    r31, r19
20 st     Z, r25
```

Listing 1: Assembly code for the previous countermeasure [19].

A novel countermeasure, which is presently used in the study, was designed based on the vulnerability causes that were analyzed. The design concept of the novel countermeasure is to apply the shuffled operation order to the confusion and diffusion layers. Therefore, a uniform hiding scheme can be applied to the entire encryption algorithm. The novel countermeasure has been experimentally proven to be safe from profiled attacks with strong attacker assumptions in an ideal environment.

1.2. Organization. The remainder of this study is structured as follows: Section 2 introduces hiding schemes, related works, and previous countermeasures as preliminaries. Section 3 describes the profiled attacks targeting the previous countermeasure and their results. Section 4 shows the novel shuffling countermeasure, which tolerates profiled attacks. Section 5 demonstrates the safety of the proposed countermeasure. Section 6 reveals the conclusion.

2. Preliminaries

This section briefly describes hiding schemes, related works, and the previous countermeasures as preliminaries.

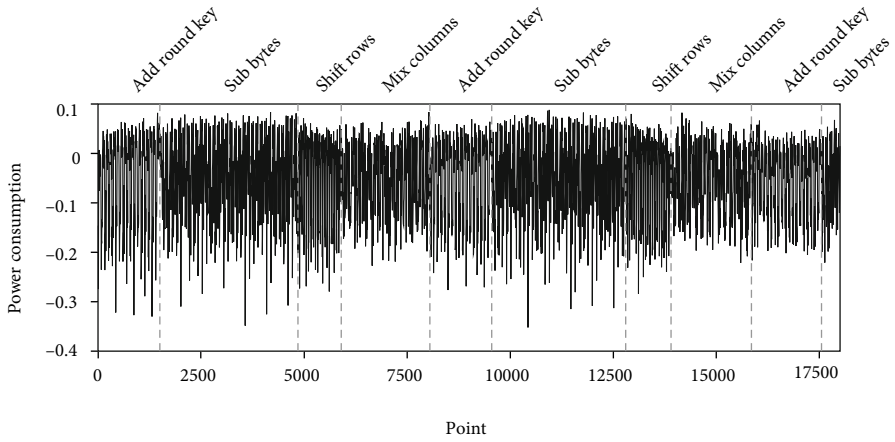


Figure 1: A power consumption trace of the previous countermeasure at optimization level -Os.

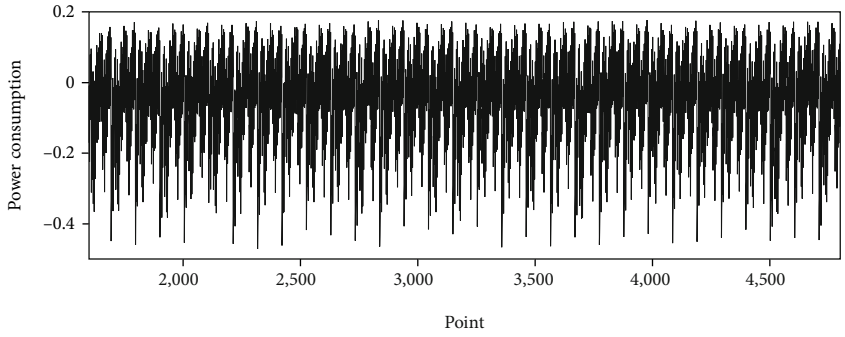


Figure 2: A power consumption trace of the first SubBytes of Figure 1.

2.1. Hiding Schemes. Hiding schemes make the power consumption of cryptographic devices independent of the intermediate values and the operations that are performed [2]. There are two approaches to achieving this purpose. The first approach is to make devices consume power randomly, and the second approach is to make devices consume the same amount of power for every operation and data value. Unfortunately, the ideal goal of randomizing or equalizing power consumption is not realistically achievable. However, there are several proposals to help get closer to this goal. These proposals are divided into two groups. The first group randomizes power consumption by performing operations at different moments. The second group touches on the amplitude dimension of power consumption. Because this study is about the first group, we would explain the first group in more detail.

The most common techniques for randomizing the execution of operations are the random insertion of dummy operations and shuffling. The random insertion of dummy operations is to randomly insert dummy operations during the execution of the operations. In this technique, randomly generated numbers are used to determine how many dummy operations to insert at different positions. As these random numbers are larger, it becomes difficult for an attacker to successfully perform an attack, but there is a disadvantage in that the implementation throughput is lowered. The shuffling randomly changes the sequence of operations that can be performed in an arbitrary order. The shuffling similarly randomizes power consumption as the random insertion of dummy operations, but the operations that can be shuffled depend on the cryptographic algorithm and is limited. Therefore, in practice, the shuffling and the random insertion of dummy operations are often combined and used.

2.2. Related Works. Assuming that shuffling and random insertion of dummy operations are combined and implemented and that up to d dummy operations can be added to n real operations, the attack complexity for recovering one key byte is $1/(n+d)$. That is, when the number of side-channel traces required to recover one key byte is α in the implementation without countermeasure, the number of required side-channel traces increases to $\alpha \times (n+d)^2$ when the countermeasures are applied [2]. However, if the attacker can distinguish dummy operations from real operations, the random insertion of dummy operations is neutralized and the number of required side-channel traces is reduced to $\alpha \times n^2$. For Sbox of Advanced Encryption Standard (AES) is fatal with a reduction of 75% when $n = d = 16$. This is a lethal number and can lower the attack complexity intended by the designer. Therefore, it is critical to make the dummy operation indistinguishable from the real operation.

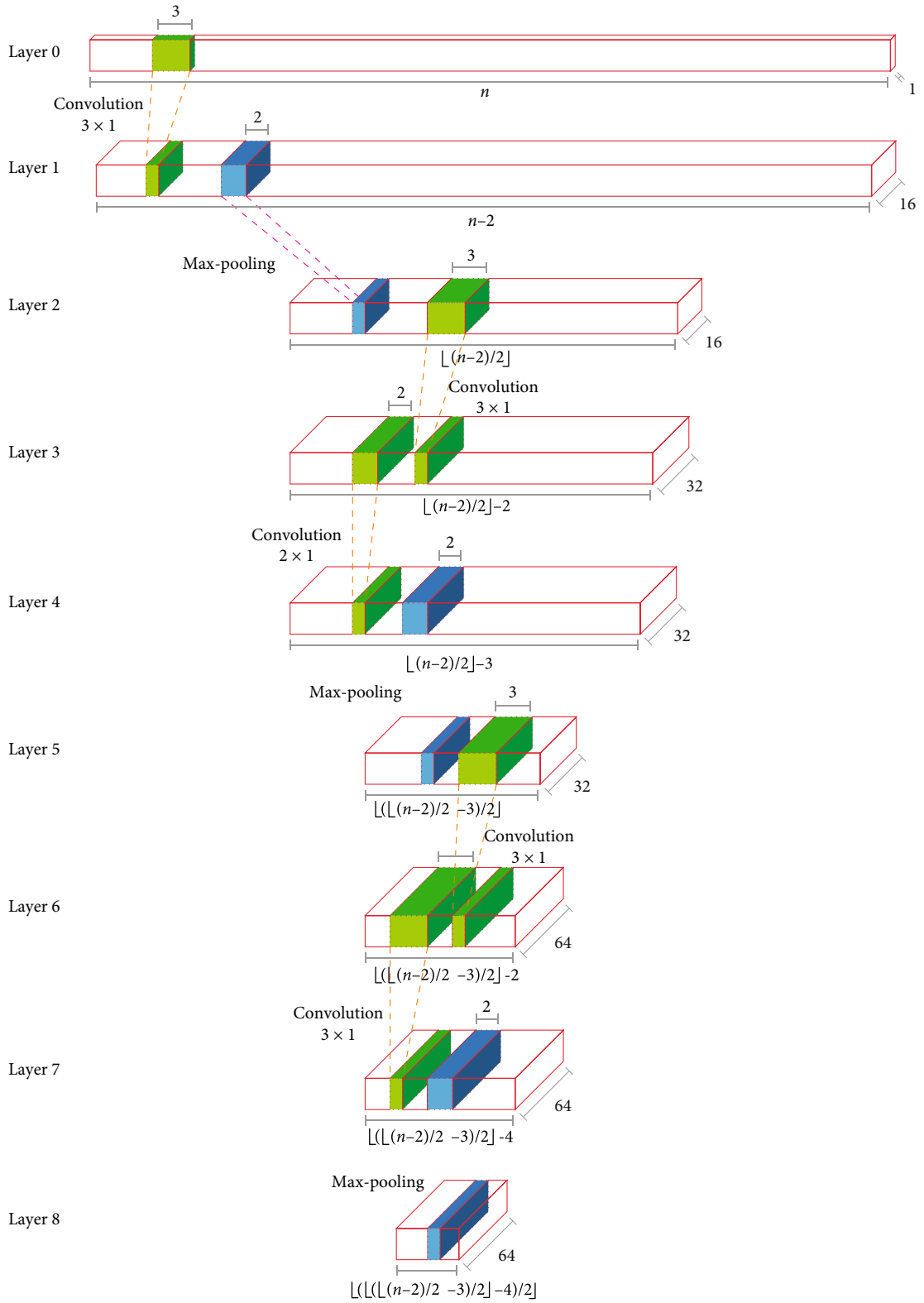


Figure 3: Convolutional neural network architecture.

Lee and Han showed for the first time the possibility of distinguishing dummy operations according to the declaration form of variables used to implement dummy operations [19]. The dummy operations were distinguished using the bounded collision detection criterion (BCDC) [20], which is a simple criterion of signal similarity. To briefly explain the method, a part of the first Sbox operation section is set as a reference area, and the BCDC value is obtained for each

area shifted by 1 point. If the first Sbox is a dummy (real) operation, the area having a low BCDC value is a part in which the dummy (real) Sbox operation is performed. As a result, the attacker can filter out dummy operations with just two trials even in a situation where the attacker does not know whether the first Sbox operation is a dummy or a real operation. However, the attack using BCDC has a disadvantage where the reference region must be selected heuristically.

The same authors later performed an attack using a neural network on the side-channel traces collected in a more noisy environment [17]. A convolutional neural network (CNN) for multilabel classification problems were used. The attacks were successful for all other declaration types except for the countermeasure proposed in their previous study [19]. However, vulnerability still exists in the countermeasure that was proposed. There is noise in the experimental environment used; thus, no vulnerability was found.

2.3. Previous Countermeasure. Lee and Han also presented a countermeasure against the proposed attack in their study [19], which proposes the attack that classifies dummy operations using BCDC. Algorithm 1 is a pseudocode of their countermeasures. It was initially thought that the vulnerabilities occur because the assembly codes are generated differently. After all, the arrays used by the dummy operations and the real operations are different, or the memory addresses referenced are different even when the same array is used. Therefore, the switch-case statements were not used to circumvent the vulnerabilities analyzed. Furthermore, the countermeasure was designed so that the dummy and the real operations can use the same array and refer directly to the array in which the shuffled order is stored.

Listing 1 is an assembly code compiled with WinAVR 20100110 (GCC-4.3.3) by implementing Algorithm 1 in the C language.

3. Profiled Attacks on Previous Countermeasure

We implemented AES with hiding schemes of Algorithm 1 on an XMEGA128D4 microprocessor [21] using C language. The power consumption was measured with a ChipWhisperer-Pro (CW1200) [18]. WinAVR 20100110 (GCC-4.3.3) is used in the compiling process and it provides -O0, -O1, -O2, -O3, and -Os as optimization levels. Detailed descriptions of each compiler optimization levels are as follows:

(i) -O0: this option does not attempt to optimize the execution time and code size. It reduces the compilation time and makes debugging generate the expected results

(ii) -O1: this compiler reduces the code size and execution time. This option only performs basic optimizations

Table 1: Test accuracies of the previous countermeasure according to optimization levels.

Optimization levels	Test accuracy
-O0	98.01875%
-O1	99.88125%
-O2	99.65000%
-O3	98.45625%
-Os	99.68125%

(iii) -O2: this compiler performs nearly all supported optimizations that do not involve a space-speed trade-off

(iv) -O3: this compiler turns on all optimizations

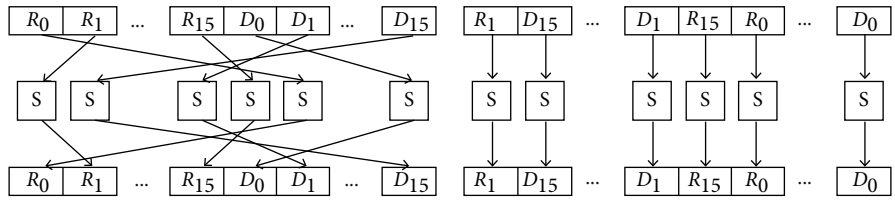
(v) -Os: this enables all -O2 optimizations, except those that often increase code size

Figure 1 shows the trace collected at optimization level -Os. It was collected to include the first two rounds and part of the third round because we unified the number of points in the trace to use the same neural network as the experiments (see Section 5). The trace for the SubBytes function of the first round is shown in Figure 2. Sixteen real and dummy operations were shuffled and performed, and it is impossible to visually distinguish whether each operation is real or the dummy.

Figure 3 shows the neural network structure to be used in the experiments in this study. The data length of the 0th layer in Figure 3 is indicated by n, which is an expression for generalization because the number of trace points varies according to the optimisation level. The number of points collected in the trace at optimisation level -O0 is 62,000, whereas the number of points collected at the rest is 18,000. This neural network uses a one-dimensional CNN, five convolution layers, and three pooling layers. ReLU is used as the activation function for each convolution layer, and batch normalization is performed after the convolution layer. The pooling layer is of max-pooling type, and a dropout ratio of 0.25 is applied after each pooling layer. After layer 8, it passes through a dense layer with 32 output nodes. The kernels of the dense layer are initialized using He normal initialization [22], and the activation function is Sigmoid. The neural network is compiled using the Adam optimizer with a learning rate of 0.001 and decay of 0.0001 and binary cross entropy as the loss function.

The attacker assumption which was set is that the attacker can collect data to train the neural network and obtain the shuffled order of operations from the profiling device. Similar to the previous study [17], the labels are composed of binary that only indicates whether the 32 operations were real or dummy. For example, if the following index of the real operations were performed:

$$[2, 3, 5, 6, 8, 10, 14, 16, 17, 18, 19, 24, 26, 27, 30, 31], \quad (1)$$



(a) In the case of the previous countermeasure (b) In the case of a novel countermeasure

Figure 4: Methods of applying the shuffling technique to the SubBytes function.

Input: Plaintext P[32] with dummy
Output: Plaintext P with dummy, shuffled orders K[32], L[32], and M[32][4], and inverse order K^{-1}

1: for $i \longleftarrow 0$ to 31 do ▷ Initialize arrays as non-shuffled orders
2: $K[i] \longleftarrow i$ and $K^{-1}[i] \longleftarrow i$ ▷ K for AddRoundKey and SubBytes
3: $S \longleftarrow (5 \times (i \bmod 16) \bmod 16) + 16 \times \lfloor i/16 \rfloor$
4: $L[i] \longleftarrow S$ and $L^{-1}[S] \longleftarrow i$ ▷ L for ShiftRows
5: for $j \longleftarrow 0$ to 3 do
6: $M[i][j] \longleftarrow \lfloor i/4 \rfloor + (i + j \bmod 4)$ ▷ M for MixColumns
7: end for
8: end for
9: for $i \longleftarrow 31$ to 1 do ▷ Shuffling
10: $R \overset{\$}{\longleftarrow} \{0, \cdots, i\}$
11: Swap P[i] and P[R]
12: Swap K[i] and K[R]
13: Swap $K^{-1}[K[i]]$ and $K^{-1}[K[R]]$
14: Swap L[i] and L[R]
15: Swap $L^{-1}[L[i]]$ and $L^{-1}[L[R]]$
16: $L[L^{-1}[i]] \longleftarrow R$, $L[L^{-1}[R]] \longleftarrow i$
17: Swap $L^{-1}[i]$ and $L^{-1}[R]$
18: Swap M[i][1] and M[R][1]
19: Swap M[i][2] and M[R][2]
20: Swap M[i][3] and M[R][3]
21: if M[i][1] = i and M[R][3] = R
22: Swap M[M[i][1]][1] and M[M[R][3]][3]
23: else
24: Swap M[M[i][1]][3] and M[M[R][3]][1]
25: end if
26: Swap M[M[i][2]][2] and M[M[R][2]][2]
27: if M[i][3] = i and M[R][1] = R
28: Swap M[M[i][3]][3] and M[M[R][1]][1]
29: else
30: Swap M[M[i][3]][1] and M[M[R][1]][3]
31: end if
32: end for
33: return P, K, L, M, K^{-1}

Algorithm 2:Generate orders for full shuffling

we can construct the following thirty-two labels:

$$[0, 0, 1, 1, 0, 1, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0]. \quad (2)$$

Here, 0 represents the dummy operation and 1 represents the real operation.

For each optimization level, 10,000 traces were collected in the variable key environment, 7,500 were used for training and 2,500 were used for validation. With a fixed key, 1,000 traces were collected for testing. The batch size was set to 10 and 100 epochs were performed; if the validation accuracy did not improve during the 10 epochs, the training was terminated early.

As a result of the training, the training was terminated early before 80 epochs in all five optimization levels and the

Input: State S[32], round key rk[32], and shuffled orders L[32] and M[4, 32]
Output: State S

1: for $i \longleftarrow 0$ to 31 do ▷ AddRoundKey
2: $S[i] \longleftarrow S[i] \oplus rk[i]$
3: end for
4: for $i \longleftarrow 0$ to 31 do ▷ SubBytes
5: $S[i] \longleftarrow Sbox[S[i]]$
6: end for
7: for $i \longleftarrow 0$ to 31 do ▷ ShiftRows
8: $T[i] \longleftarrow S[L[i]]$
9: end for
10: for $i \longleftarrow 0$ to 31 do ▷ MixColumns
11: $temp1 \longleftarrow T[M[i][1]] \oplus T[M[i][2]] \oplus T[M[i][3]]$
12: $temp2 \longleftarrow xtime(T[i] \oplus T[M[i][1]])$
13: $S[i] \longleftarrow temp1 \oplus temp2$
14: end for
15: return S

Algorithm 3: Shuffled round function of AES

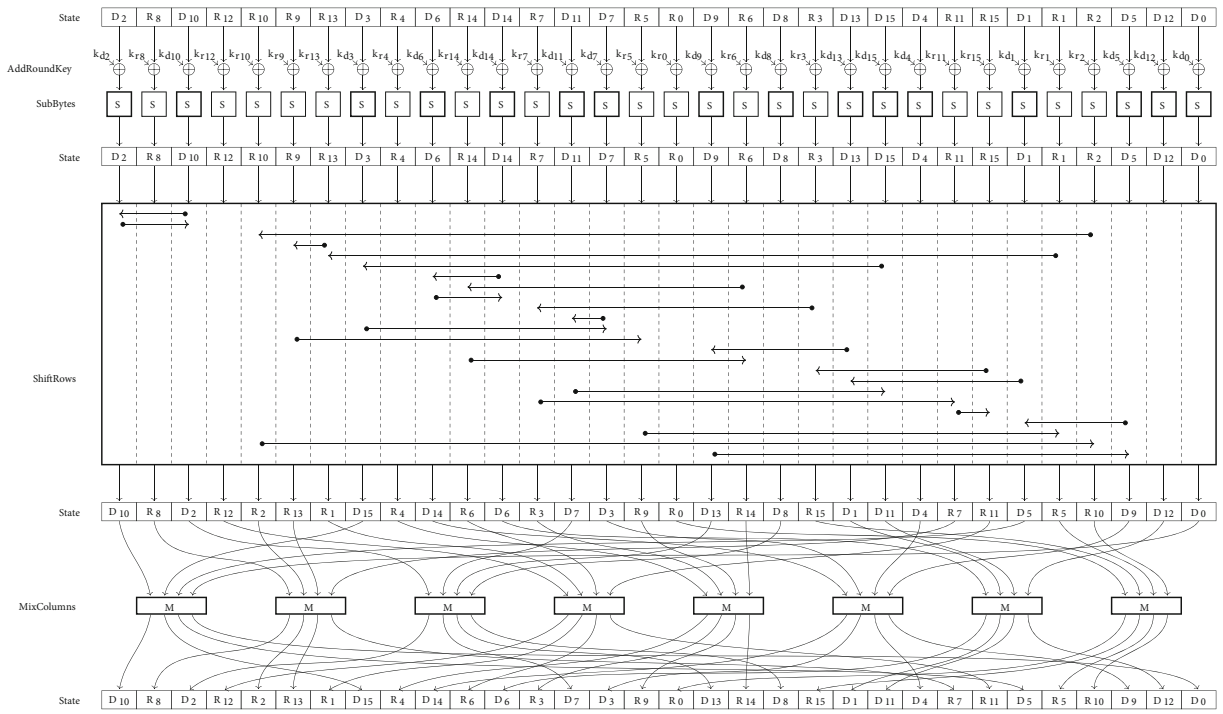


Figure 5: Structure of the novel shuffling countermeasure.

```
1;   OUT[i] = Sbox[IN[i]]
2    movw    r26, r22
3    movw    r28, r24
4    ld      r30, Y+
5    movw    r24, r28
6    ldi     r31, 0x00
7    subi    r30, 0xF6
8    sbci    r31, 0xDF
9    ld      r20, Z
```

Listing 2: Assembly code for the novel countermeasure.

validation accuracies were over 98%. There was no overfitting, and test accuracy at the optimization levels is shown in Table 1.

4. Novel Shuffling Countermeasure

In this section, we highlight the problem of the previous countermeasure and proposed a novel shuffling countermeasure that is safe from profiled attacker's assumption.

4.1. Motivation. Previous research has shown that the compiled assembly codes are different when the variables used by the real and dummy operations are declared separately, and the assembly codes are different when the switch-case

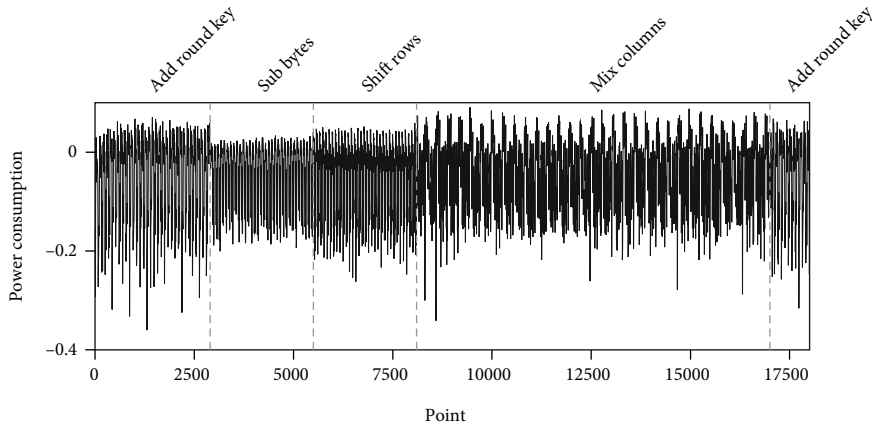


Figure 6: A power consumption trace of the novel countermeasure at optimization level -Os.

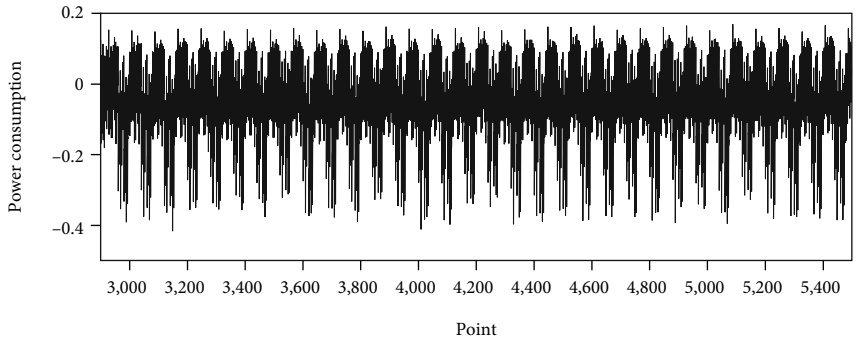


Figure 7: A power consumption trace of the first SubBytes of Figure 6.

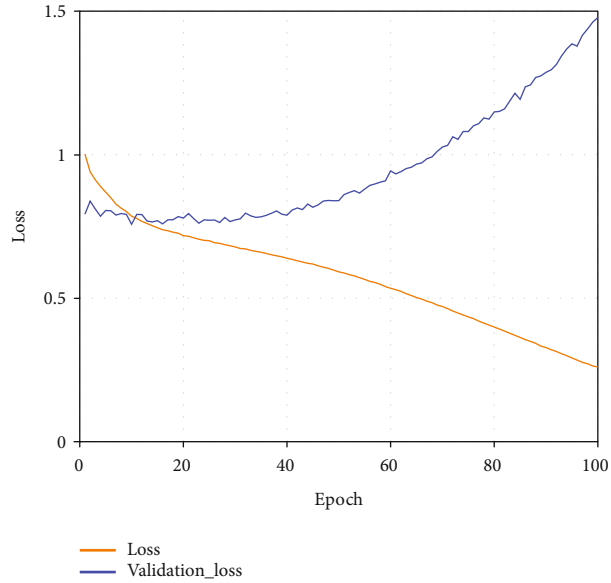
statement is used even when the same variable is used [19]. Therefore, similar to Algorithm 1, the authors made the real and dummy operations use the same array and operated by directly referencing the array in which the shuffled order is stored without using a switch-case statement. Therefore, the assembly codes for the real and dummy operations are generated identically as shown in Listing 1. Based on this, the authors assumed that their countermeasure was safe.

Their experiment has a limitation in that the experimental verification was conducted on a white card with many noises compared to the ideal environment. Because of the low signal-to-noise ratio in noisy environments, leakage is obscured by noise, making accurate experimental verification difficult. Therefore, because of the experimental safety verification of the previous countermeasure in a low-noise environment, it was not safe at all optimization levels (see Table 1).

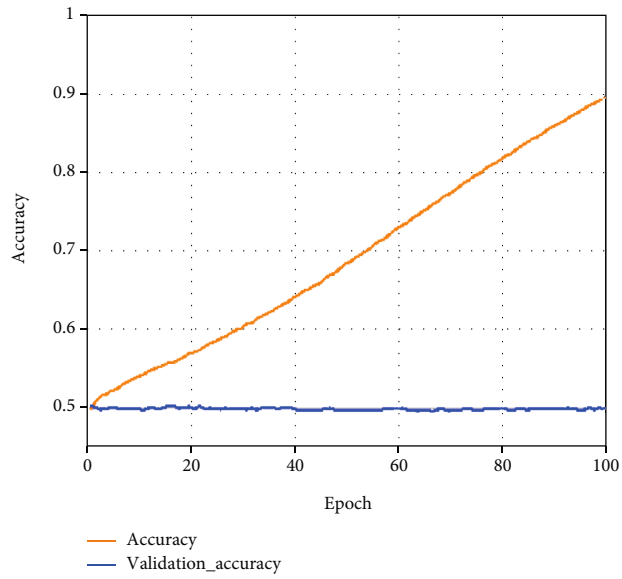
To analyze the cause of this vulnerability, the assembly code of Listing 1 was noted. Lines 3 and 4 of Listing 1 store the address of the ORD array to the r26 and r27 registers. Then, in line 7, the value of the element of the currently pointed ORD array is loaded into the r18 register, and the address of the following element is pointed. It was assumed that the attacker could see the shuffled order (see Section 3). Therefore, line 7, in which the value of shuffled order is stored in the register, is a vulnerable cause. The traces were collected by deleting line 7 and attempted to attack the collected traces, which yielded a test accuracy of nearly 50%. Therefore, this demonstrated that line 7 is a vulnerable cause.

This study concluded that the method of applying shuffling should be modified to eliminate the abovementioned vulnerability. A schematic of the method for applying the shuffling scheme to the SubBytes function of the previous countermeasure is shown in Figure 4(a). The values for real and dummy operations are sequentially stored in the array, and the operations are performed regardless of the order stored in the array by referring to the shuffled order. Because the vulnerability exists in the section that refers to the shuffled order, it is necessary to configure the plaintext to be stored in the shuffled order from the start (see Figure 4(b)). However, it is difficult to perform encryption while storing the values in the array in the shuffled order. This is because, among the internal operations of the AES encryption algorithm, ShiftRows and MixColumns are byte-dependent operations. AddRoundKey and SubBytes, which are byte-independent operations, can sequentially operate on the shuffled and stored state array, but byte-dependent operations must calculate the indices of the values to be operated together. This will be covered in more detail in the following section.

4.2. The Expansion of Shuffling Scheme to Other Operations. In this section, a method was proposed for expanding the shuffling scheme used in SubBytes to ShiftRows and MixColumns, which are byte-dependent operations of AES.



(a) Loss



(b) Accuracy

Figure 8: Training result of the novel countermeasure with optimization level -Os.

Table 2: Test accuracies of the novel countermeasure according to optimization levels.

Optimization levels	Test accuracy
-O0	49.56250%
-O1	50.25625%
-O2	50.23750%
-O3	50.60625%
-Os	49.93125%

Algorithm 2 is a pseudocode that shuffles plaintext array and order arrays for AddRoundKey, SubBytes, ShiftRows, and MixColumn. In this algorithm, 16 dummy operations are used. P is the plaintext array with dummy added, and K, L, and M are the order arrays of AddRoundKey and SubBytes, ShiftRows, and MixColumns, respectively. Additionally, a reverse-order array K^{-1} of K is generated for the recovery of the ciphertext. Only the order array M for MixColumn, where four bytes are used for operation at once, is a two-dimensional array with a size of 32×4, and the rest are all one-dimensional arrays with a length of 32.

First, lines 1 to 8 initialize the arrays in order before applying the shuffling scheme. After that, lines 9 to 32 apply the shuffling scheme using Fisher-Yates shuffle [23]. This loop from the highest index to the lowest, and lines 11 to 13 shuffle the plaintext array and the order array for AddRoundKey and SubBytes, respectively. Lines 14 and 17 shuffle the order array for ShiftRows, which is different from

Table 3: Cycles per byte of AES implementations with and without a countermeasure.

Implementation	Cycles per byte
Unprotected AES	127
AES with the previous countermeasure	572
AES with the novel countermeasure	1028

order K for byte-independent operation, since values referring to the swapped value must also be swapped, so line 17 is required. Lines 18 to 31 should be shuffled in the order of MixColumns. In the case of M, because it is a two-dimensional array and four bytes are used in the operation immediately, a maximum of 12 values must be swapped.

Algorithm 3 is the pseudocode for the round function that uses the shuffled orders generated in Algorithm 2. AddRoundKey and SubBytes are byte-independent operations, and because the state array has already been shuffled and stored, sequential operations are performed in the order in which they are stored. ShiftRows is a one-byte dependent operation that operates regarding the order array L (line 8). In MixColumns, the operations from lines 11 to 13 are conducted. The operation on one column of MixColumns is a polynomial product with a fixed polynomial $m(x)$ using $x^4 + 1$ as the multiplied modulo over GF(2^8), which is given by

$$m(x) = \{03\}x^3 + \{01\}x^2 + \{01\}x + \{02\}. \tag{3}$$

Formulating this in terms of the coefficient is as follows:

$$s'_i = \{02\} \cdot s_i \oplus \{03\} \cdot s_{(i+1) \bmod 4} \oplus s_{(i+2) \bmod 4} \oplus s_{(i+3) \bmod 4}. \tag{4}$$

This can be written as follows:

$$s'_i = \left(s_{(i+1) \bmod 4} \oplus s_{(i+2) \bmod 4} \oplus s_{(i+3) \bmod 4}\right) \oplus \{02\} \cdot \left(s_i \oplus s_{(i+1) \bmod 4}\right). \tag{5}$$

The codes from lines 11 to 13 of Algorithm 3 are the same as Equation (5). Here, $x$time means $\{02\} \cdot x$.

Figure 5 schematically shows Algorithm 3. The index of the state array is an example, and arrows in the schematic of ShiftRows and MixColumns change each time according to shuffling.

5. Demonstration

In this section, the safety of the novel shuffling countermeasure proposed in Section 4 is experimentally confirmed. First, the assembly code generated by the compiler was observed. Listing 2 is the assembly code of the SubBytes part of the novel countermeasure. Sbox operations are performed in the order stored in the state array without referring to the order array.

Figure 6 shows the power consumption trace of the novel countermeasure. Compared with Figure 1, the length of one round is approximately 2.1 times longer. In detail, the length is increased by applying the dummy operation and shuffling scheme to AddRoundKey, ShiftRows, and MixColumns, whereas the length of SubBytes is shortened by reducing the array reference once. The power consumption trace of the SubBytes part is shown in Figure 7. It is impossible to distinguish between the dummy and the real operation.

The traces of the novel countermeasure were also collected in the same environment as the traces of the previous countermeasure. The power consumption trace of the XMEGA128D4 chip was collected with ChipWhisperer-Pro (see Section 3). At each of the four optimization levels, 10,000 traces were collected with the variable key and 1,000 traces were collected with the fixed key. Additionally, the artificial neural network also used the same model used in the previous countermeasure (see Figure 3). The experiments were conducted using the same learning environment in Section 3. The learning graph at the optimization level -Os is shown in Figure 8. The learning graphs at the remaining optimization levels also have a similar shape to Figure 8. While the training loss decreases and the training accuracy rises, the validation loss increases again and the validation accuracy stops around 0.5. The test accuracy of the novel countermeasure at all optimization levels is shown in Table 2. The accuracy of 0.5 indicates that the neural network does not properly classify because it is a binary classification problem. Therefore, our novel countermeasure is safe at all optimization levels because attackers cannot distinguish between real and dummy operations.

The countermeasure of the study is not only safe but also effective. The cycles per byte of AES implementations are shown in Table 3. The implementation with the novel countermeasure takes approximately eight times more cycles per byte than the unprotected implementation, and approximately 1.8 times as long as the implementation with the previous countermeasure applied. The previous countermeasure used dummy operations for only the SubBytes function, but the novel countermeasure used dummy operations for all functions, so the overhead is inevitable. According to H. Kim et al. [24], when a masking countermeasure is applied to AES, the first and second masking takes approximately 1.7 times and 23.7 times more cycles per byte, respectively, compared to the nonprotected implementation. Considering this, the cost of adding dummy operations and applying shuffling to the entire encryption process, which is eight times more cycles, is tolerable.

6. Conclusions

In this study, the authors questioned the safety of the previous shuffling countermeasure. In a previous study [17], the authors designed the countermeasure to be safe against attackers using machine learning and performed experimental verification but were unable to fully confirm the countermeasure because the experimental environment was set to a noisy environment. As a result of reverification in the appropriate environment, it was confirmed that the previous countermeasure was not safe.

Previous countermeasures applied the hiding schemes only to the byte-independent operations of the cryptographic algorithm. The cause of the weakness of the previous countermeasure was analyzed, and a novel countermeasure was designed using the shuffling and random insertion of dummy operations schemes up to the byte-dependent operation of AES to avoid the weakness. Moreover, to confirm the safety of the proposed countermeasure, an experimental verification was conducted in an environment with as little noise as possible. As a result, even assuming a strong attacker who knows the indices of the dummy operations, the neural network has not learned to distinguish the dummy operations. Therefore, the proposed countermeasure is safe.

In future works, countermeasures for other block ciphers can be designed similarly to those designed for AES in this paper. Since the types of components used for each block cipher are different, a dedicated design for each is required. In this paper, an optimized design of the proposed countermeasure was not considered. Therefore, an optimized design for the full-hiding scheme is also required.

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 there is no conflict of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2021-0-00903, Development of physical channel vulnerability-based attacks and its countermeasures for reliable on-device deep learning accelerator design).

References

[1] Cisco, *Cisco annual internet report (2018-2023)*, 2020, https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internetreport/white-paper-c11-741490.html.

[2] S. Mangard, E. Oswald, and T. Popp, *Power Analysis Attacks: Revealing the Secrets of Smart Cards*, vol. 31, Springer Science & Business Media, 2008.

[3] P. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," in *Annual international cryptology conference*, pp. 388–397, Berlin, Heidelberg, 1999.

[4] E. Brier, C. Clavier, and F. Olivier, "Correlation power analysis with a leakage model," in *International Workshop on Cryptographic Hardware and Embedded Systems*, pp. 16–29, Springer, Berlin, Heidelberg, 2004.

[5] C. Herbst, E. Oswald, and S. Mangard, "An AES smart card implementation resistant to power analysis attacks," in *International conference on applied cryptography and network security*, pp. 239–252, Berlin, Heidelberg, 2006.

[6] M. Rivain and E. Prouff, "Provably secure higher-order masking of AES," in *International Workshop on Cryptographic Hardware and Embedded Systems*, pp. 413–427, Springer, Berlin, Heidelberg, 2010.

[7] M. Rivain, E. Prouff, and J. Doget, "Higher-order masking and shuffling for software implementations of block ciphers," in *International Workshop on Cryptographic Hardware and Embedded Systems*, pp. 171–188, Springer, Berlin, Heidelberg, 2009.

[8] J. G. van Woudenberg, M. F. Witteman, and B. Bakker, "Improving differential power analysis by elastic alignment," in *Cryptographers' Track at the RSA Conference*, pp. 104–119, Berlin, Heidelberg, 2011.

[9] D. Strobel and C. Paar, "An efficient method for eliminating random delays in power traces of embedded software," in *International Conference on Information Security and Cryptology*, pp. 48–60, Berlin, Heidelberg, 2011.

[10] F. Durvaux, M. Renauld, F. X. Standaert, L. V. Oldeneel tot Oldenzeel, and N. Veyrat-Charvillon, "Efficient removal of random delays from embedded software implementations using hidden Markov models," in *International Conference on Smart Card Research and Advanced Applications*, pp. 123–140, Berlin, Heidelberg, 2012.

[11] L. Wu and S. Picek, "Remove some noise: on pre-processing of side-channel measurements with autoencoders," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, vol. 2020, no. 4, pp. 389–415, 2020.

[12] Y.-S. Won, X. Hou, D. Jap, J. Breier, and S. Bhasin, "Back to the basics: seamless integration of side-channel pre-processing in deep neural networks," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 3215–3227, 2021.

[13] D. Kwon, H. Kim, and S. Hong, "Non-profiled deep learning-based side-channel preprocessing with autoencoders," *IEEE Access*, vol. 9, pp. 57692–57703, 2021.

[14] H. Maghrebi, "Assessment of common side channel countermeasures with respect to deep learning based profiled attacks," in *2019 31st International Conference on Microelectronics (ICM)*, pp. 126–129, Cairo, Egypt, 2019.

[15] H. Maghrebi, *Deep Learning Based Side Channel Attacks in Practice*, Cryptology ePrint Archive, 2019.

[16] L. Masure, C. Dumas, and E. Prouff, "A comprehensive study of deep learning for side-channel analysis," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, vol. 2020, no. 1, pp. 348–375, 2020.

[17] J. Lee and D.-G. Han, "DLDDO: deep learning to detect dummy operations," in *International Conference on Information Security Applications*, pp. 73–85, Cham, 2020.

[18] NewAE, *CW 1200: Chipwhisperer-pro*, NewAE Technology, 2021, https://media.newae.com/datasheets/NAE-CW1200_datasheet.pdf.

[19] J. Lee and D.-G. Han, "Security analysis on dummy based side-channel countermeasures–case study: AES with dummy and shuffling," *Applied Soft Computing*, vol. 93, p. 106352, 2020.

[20] I. Diop, P.-Y. Liardet, Y. Linge, and P. Maurine, "Collision based attacks in practice," in *2015 Euromicro Conference on Digital System Design*, pp. 367–374, Madeira, Portugal, 2015.

[21] Atmel, *AVR XMEGA D4 Devices Datasheet*, Atmel Corporation, 2021, http://ww1.microchip.com/downloads/en/

DeviceDoc/Atmel-8135-8-and-16-bit-AVRmicrocontroller-ATxmega16D4-32D4-64D4-128D4_datasheet.pdf.

[22] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: surpassing human-level performance on ImageNet classification," in *Proceedings of the IEEE international conference on computer vision*, pp. 1026–1034, Santiago, Chile, 2015.

[23] R. Durstenfeld, "Algorithm 235: random permutation," *Communications of the ACM*, vol. 7, no. 7, p. 420, 1964.

[24] H. Kim, S. Hong, and J. Lim, "A fast and provably secure higher order masking of AES S-Box," in *International Workshop on Cryptographic Hardware and Embedded Systems*, pp. 95–107, Springer, Berlin, Heidelberg, 2011.

Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 4408987, 17 pages
https://doi.org/10.1155/2022/4408987

Research Article

Birds of a Feather Flock Together: Generating Pornographic and Gambling Domain Names Based on Character Composition Similarity

Yanan Cheng,[1] **Hao Jiang**,[1] **Zhaoxin Zhang**,[1] **Yuejin Du**,[2] **and Tingting Chai**[1]

[1]*Faculty of Computing, Harbin Institute of Technology, Harbin 150001, China*
[2]*Beijing Qihoo Technology Co., Ltd, Beijing 100015, China*

Correspondence should be addressed to Zhaoxin Zhang; zhangzhaoxin@hit.edu.cn and Tingting Chai; ttchai@hit.edu.cn

Received 10 May 2022; Revised 5 June 2022; Accepted 23 June 2022; Published 11 July 2022

Academic Editor: Yuanlong Cao

Cybercriminals often register many pornographic or gambling domains (known as abusive domains) with similar character compositions in bulk to reduce their investment in buying domains and make it easier for clients to remember and spread them. Therefore, this study combines the ideas of text similarity and text generation and proposes an abusive domain generation model based on GRU for rapidly generating new abusive domain names from known ones. Additionally, we develop a two-layer detection system for pornography and gambling domains using fastText and CNN models to obtain an abusive domain dataset for model training and validation. In the end, our detection system identifies pornographic and gambling domains with 99% precision while balancing correctness and speed. By inputting 40,000 random keywords into the abusive domain generation model, we obtained 130,220 online domains that served web pages, of which about 66% were pornographic or gambling domains. The results show that by exploiting cybercriminals' behaviors in registering abusive domain names, such as bulk registration of similar domain names, we can prospectively acquire a large number of new abusive domains based on known ones. This study demonstrates that predicting new abusive domains not only expands the domain blacklist but also allows researchers to target the generated suspicious domains and dispose of them in time before they show abusive behavior.

1. Introduction

Cybercriminals are establishing more and more pornographic and gambling domains (or websites, collectively referred to as abusive domain names) in pursuit of profit. At the same time, with the growth of the Internet and social media, people are increasingly exposed to these abusive domain names, either intentionally or unintentionally, including children and minors. Pornographic videos and images hurt the physical and mental health of minors. Many gambling sites are fraudulent sites that cheat people out of their money [1]. At the same time, the current state of the global epidemic of COVID-19 has led to an even more rampant spread of pornographic and gambling domains on the Internet [2–7]. Therefore, the sooner governments, security institutions, and Internet entities can discover, block, and handle these pornographic and gambling domains, the more they can mitigate the harm caused by these domains [8]. Therefore, from a technical perspective, the first significant challenge for each Internet entity is how to quickly, accurately, and early discover pornography and gambling domains, which is the research objective of this paper.

Generally, much of the existing research in pornography and gambling domain discovery focuses on detection, where the website (domain) is entered into the detection model. Then, information about the website (e.g., text or images)

is used to determine whether the domain is pornographic or gambling. These detection methods are necessary to discover pornographic and gambling domains. However, detection methods cannot discover abusive domains earlier because they can only detect the domains that are entered into the models. In order to discover the abusive domains earlier, we need to adopt a new perspective to start with.

Through our empirical analysis of many pornographic and gambling domain names, we find that there are similarities in the composition of these domain names. These similar characteristics are mainly reflected in two aspects. On the one hand, to facilitate abusive domain management and memorization, cybercriminals register many domain names with similar compositions, such as *porn[0-9].com*, in bulk. On the other hand, many pornographic and gambling domain names have no special meaning but are just combinations of letters and digits. Because domains with meaningful word combinations are expensive to register, cybercriminals register many domain names with meaningless compositions in bulk to reduce the investment in malicious attack activities (Section 2).

Therefore, in this paper, we develop a two-layer detection system for pornography and gambling domains using fastText and CNN models (Section 3.1), which is able to identify abusive domains quickly and accurately. Meanwhile, using the compositional similarity features of many pornographic and gambling domains, we combine the ideas of text similarity and text generation and propose a novel abusive domain generation model based on GRU to generate new pornographic and gambling domains from existing ones (Section 3.2 and Section 3.3). Finally, our detection system identifies pornographic and gambling domains with 99% precision while balancing correctness and speed. By inputting 40,000 random keywords into the abusive domain generation model, we obtained 130,220 online domains that served web pages, of which about 66% were pornographic or gambling domains (Section 4).

In short, we make the following contributions:

(i) We develop a two-layer detection system using fastText and CNN models to identify pornographic and gambling domains. The system is capable of ensuring high detection efficiency while maintaining a high detection accuracy rate for abusive domain names. In addition, this method can exclude websites that contain only pornographic or gambling keywords in the text of the page

(ii) For the first time, using existing abusive domains, we propose a novel approach to generate many new and undiscovered abusive domains based on domain composition similarity. This method enables us to discover many pornographic and gambling domains earlier so that they can be blocked and handled in a timely manner

(iii) For the first time, we share a database (https://reurl.cc/0p27db, accessed on 6 May 2022, access password: nist@HIT) of manually labeled website snapshots of abusive domains, containing 18,428 pornography domains and 15,578 gambling domains. We hope that more security communities and researchers can use these samples for research on pornography and gambling domain detection or generation

In summary, this paper aims to discover a large number of pornographic and gambling domains as quickly, accurately, and early as possible. This paper is intended for audiences across Internet infrastructure, cybersecurity industries, and researchers.

2. Background and Related Work

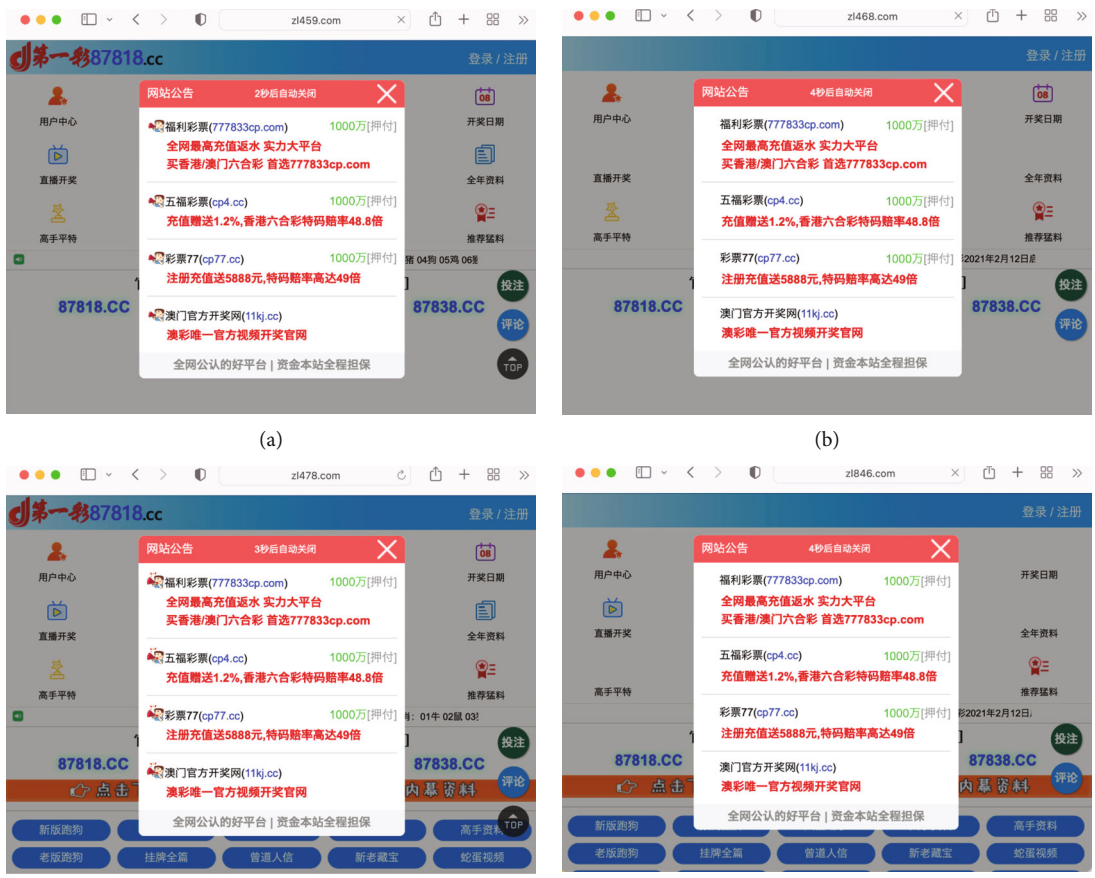
2.1. Background

2.1.1. Similarity in the Composition of Abusive Domains. One of the fundamental assumptions of this study is that a substantial number of abusive domain names share a common character composition, i.e., they follow the same composition rules. We provide some examples of domain names with similar character compositions. As shown in Figure 1, the four gaming domains display the exact same web page content, and their domain name character composition rules conform to the rules *zl* ∗∗∗ *.com*. The same situation exists for pornographic domains, as shown in Figure 2, which all conform to rule ∗∗∗∗ *av.com*.

In addition, we checked the domain name certificate of the gambling domain *lh1769.com*, as shown in Figure 3, and found 120 domain names with similar character composition to this gambling domain. Once again, it is proven that pornography or gambling sites use a large number of domain names with similar character composition. This case not only facilitates distribution but also makes it easy for the viewer to remember the domains of sites, and when a domain name is not available to access the site, the viewer uses a new domain name to access it.

By observing many abusive domain names of pornography and gambling types, we found two main characteristics of these abused domain names. First, they are mainly composed of pure numbers or a mixture of numbers and letters with no real meaning. Second, the similarity is primarily shown by the fact that some characters (numbers or letters) in the domain name stay the same, but many other characters in their next or previous positions change. Therefore, we can design methods to discover abusive domain names of similar composition as soon as possible using these characteristics.

On the other hand, many pornography and gambling domains consist of meaningless letters and numbers, as described above. Figure 4 shows the frequency of letters and digits at various positions in popular domains (Alexa top 1 million, http://s3-us-west-1.amazonaws.com/umbrella-static/top-1m.csv.zip, accessed on 6 May 2022) and abusive domains (domain length less than 16). We can find that the frequency of characters in abused domain names is different from popular domain names, especially a large number of numeric characters that appear in pornography and gambling domain names.



(a) (b) (c) (d)

Figure 1: Gambling domain names with similar character composition. (a) Domain *zl459.com*. (b) Domain *zl468.com*. (c) Domain *zl478.com*. (d) Domain *zl846.com*.

In general, the similarity in character composition and nonsense of many pornography and gambling domain names provides a practical basis for generating new abusive domain names.

2.1.2. Abusive Domains in Disguise. Pornography and gambling websites have apparent textual features, such as many keywords (https://github.com/mrcheng0910/reporting_abusive_domains/blob/main/abusive_keywords.txt, accessed on 6 May 2022) related to pornography or gambling. Therefore, high detection accuracy can be achieved by designing a text-based classifier. Yang et al. designed and implemented an SVM-based classifier to achieve 99% accuracy in detecting online gambling websites [1]. Therefore, we refer to text-based related methods to filter gambling and pornographic websites from the textual perspective.

On the other hand, miscreants from online underground economies regularly exploit website vulnerabilities and inject fraudulent content into web pages to promote illicit goods and services. Adversaries often manage to inject content stealthily by obfuscating the description of illegal products and/or the presence of defacements to make them undetectable [9]. As shown in Figure 5, gambling-related keywords are maliciously embedded in the title, description, and keyword tags of the normal website, respectively. However, the page displayed to the users in the browser is benign.

As a result, such sites are easily misclassified as abusive domain names through text-based classifiers. In view of this situation, this paper implements an image-based abusive domain name detection tool in addition to developing a text-based filter. The text-based filter is fast, consumes fewer resources, and can filter out abusive domain names from a large number of websites as quickly as possible. The image-based classifier further detects the filtered abusive domain names to improve the final detection accuracy. In this paper, we use convolutional neural networks (CNNs) to detect website snapshots to find pornographic or gambling domains. We describe both methods in detail in Section 3.1.

2.2. Related Work

2.2.1. Abusive Domain Detection. Srinivasan et al. [10] present DeepURLDetect (DURLD), a method that extracts features from character level embedding using hidden layers in deep learning architectures and then uses a nonlinear activation function to predict the likelihood of the URL is malicious or not. Lison et al. [11] established a model for detecting domain generation algorithm (DGA) domains using recurrent neural networks. This model was capable of making predictions only based on domain names, without the need for human participation or access to external

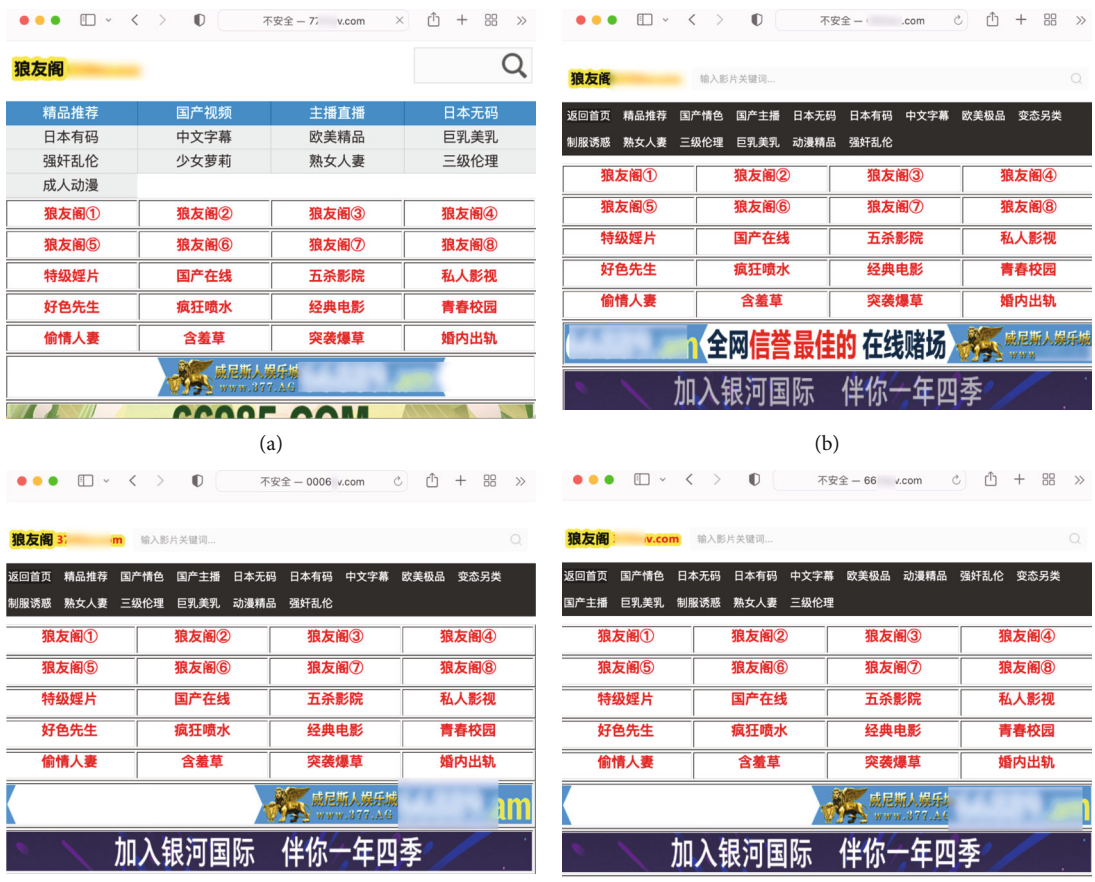


Figure 2: Pornographic domain names with similar character composition. (a) Domain *7757av.com*. (b) Domain *0004av.com*. (c) Domain *0006av.com*. (d) Domain *6687av.com*.

Figure 3: Multiple abusive domain names share one certificate. (a) Domain *lh1769.com*. (b) List of domains that use the domain *lh1769.com*'s certificate.

resources. Curtin et al. [12] provided a novel machine learning system built partially on recurrent neural network (RNN) that is capable of classifying DGA-generated domain names even from families traditionally understood as difficult. Xu et al. [13] proposed a novel n-gram combined character-based domain classification (n-CBDC) model using n-grams and a deep convolutional neural network. This model operates end-to-end and does not require manually extracted features or DNS context information; it only requires the domain name itself as input and can

Figure 4: The frequency of characters at each position in the popular and abusive domains.

Figure 5: Gambling-related keywords are maliciously embedded in the benign website.

automatically assess the probability that the domain name was formed using DGAs. Bharathi et al. [14] proposed to take a string of characters as the input given in the domain names and classify them as either benign or malicious domain names using deep learning architectures such as Long-Short-Term Memory (LSTM) and bidirectional LSTM. Ren et al. [15] applied a deep neural network model with an attention mechanism (ATT-CNN-BiLSTM) for the detection and classification of DGA domain names. The main thought behind their ensemble model is that the validity of the context inherent in domains could contain sufficient information with which to distinguish DGA domain names, especially the wordlist-based ones. The majority of traditional approaches focus on a particular feature of these pornographic and gambling websites, which leaves out more nuanced and problematic scenarios. Chen et al. [16] developed an automatic detection system for pornographic and gambling websites based on visual and textual content using a decision process to address this issue. Similarly, Zhao et al. [17] proposed Porn2Vec, a robust end-to-end framework for detecting pornographic websites using contrastive learning. This framework, in particular, models pornographic websites as a heterogeneous network composed of websites, web pages, images, and text, as well as their interaction relationships, and formalizes pornographic website identification as a node classification problem on the graph. Additionally, the model employs a novel contrastive learning-based heterogeneous graph embedding method to learn the high-level representation of web pages by combining image-based, text-based, and structure-based information

concurrently. Finally, the learned website characteristics are sent into a neural network to train an automatic pornographic website detection model.

2.2.2. Prediction or Generation Based on RNN. RNN is one of the most promising tools for deep learning, which has been applied to speech recognition, machine translation, music generation, and text generation in a large number of previous studies [18–24]. For the first time, we applied it to domain name generation based on the idea of text generation.

The RNN is the first algorithm that remembers its input, due to an internal memory, which makes it perfectly suited for machine learning problems that involve sequential data. Therefore, many studies use RNN for prediction or generation tasks. Wang et al. [18] proposed a novel attention-based LSTM [25] model for song iambic generation. Specifically, they encoded the cue sentences by a bidirectional LSTM model and then predicted the entire iambic with the information provided by the encoder, in the form of an attention-based LSTM that can regularize the generation process by the fine structure of the input cues. Sturm et al. [19] applied the LSTM model to music transcription modeling and composition. They built and trained the LSTM network using approximately 23,000 music transcriptions expressed using a high-level vocabulary (ABC notation) and then used them to generate new transcriptions. For the purpose of generating Chinese classic poetry, Luo et al. [21] introduced a novel text steganography technique based on the RNN encoder-decoder structure. They employed a keyword to construct the first line of a quatrain and then generated the subsequent lines one by one using the LSTM model. Additionally, they used a template-based generating method and established a word-choosing strategy based on inner-word mutual knowledge to combat poetry's dramatic decline in quality. Accurate and real-time traffic flow prediction is important for traffic control. Fu et al. [22] used LSTM and gated recurrent units (GRU) neural network methods to predict short-term traffic flow. Unlike prior template-based systems, Liu et al. [23] showed a system for generating Chinese classical poetry dubbed Deep Poetry that utilizes neural networks trained on over 200 thousand poems and 3 million pieces of ancient Chinese prose. This technology can generate Chinese classical poetry from plain text, images, or aesthetic notions. More importantly, this method allows users to engage in the process of composing poetry. Bartoli et al. [24] proposed and assessed a system for automatically generating restaurant reviews suited to the desired rating and restaurant category using LSTM. They trained the neural network on a set of authentic restaurant reviews in order to produce text that appears to be a restaurant review.

To summarize, the numerous existing approaches for detecting abusive domains listed above each rely on a single type of feature, such as the domain character, the URL, textual, or visual features. In comparison to these single-feature detection methods, hybrid feature-based methods perform better and offer broader development prospects. Therefore, this study combined the textual and visual features of the website to detect gambling and pornographic domains. Additionally, many of the different types of tasks (e.g., classical poetry and criticism) predicted or generated are carried out using the RNN model and perform well. Therefore, we generate new abusive domain names based on the RNN model.

In particular, because the purpose of the research described in this paper is to generate or predict new abusive domain names based on existing abusive domain names, the accuracy of detecting abusive domain names should be high enough. In addition, considering the significant resource and time consumption of image-based detection, therefore, we first filter out many suspected gambling and pornographic domains with a text-based detection method and then use an image-based approach for further verification. In this way, the accuracy and efficiency of domain name abuse detection meet the requirements.

3. Methodology

In this section, we design methods for generating more new abusive domains based on the abusive domain samples that have been acquired, as shown in Figure 6. Thus, our method consists of three major stages: acquiring abusive domain name samples; clustering abusive domain names based on similar composition rules; generating new abusive domain names based on these clusters.

3.1. Obtaining Abusive Domains. As shown in Figure 6, the work in this stage is mainly to build a database of abusive domains for generating new abusive domain names. This stage contains three main tasks: one is to obtain the web content of a large number of domain names, including HTML source code and snapshots; two is to detect pornographic and gambling domains based on HTML source code and snapshots, respectively; three is to discover more abusive domain names based on the certificate features of pornographic and gambling websites.

3.1.1. Crawling Web Content. First, we downloaded over 260 million domain names from Domain Monitor [26]. These domains come from 1500 zones, which indicates that these domains have DNS records. Second, we developed two types of web crawlers to crawl web content.

(i) Requests-based web crawler. This crawler uses the Python-requests [27] package to crawl the HTML source code of domains. Then, we extract the title, keywords, and description tags from the source code. This text information is used to determine if the domain name is pornographic or gambling

(ii) Selenium-based web crawler. This crawler uses the Selenium webdriver [28] to get the snapshots of the specific domains. We use these snapshots to detect pornographic and gambling domains. Compared to fetching web page source code, fetching web page snapshots is slower and consumes more computing resources

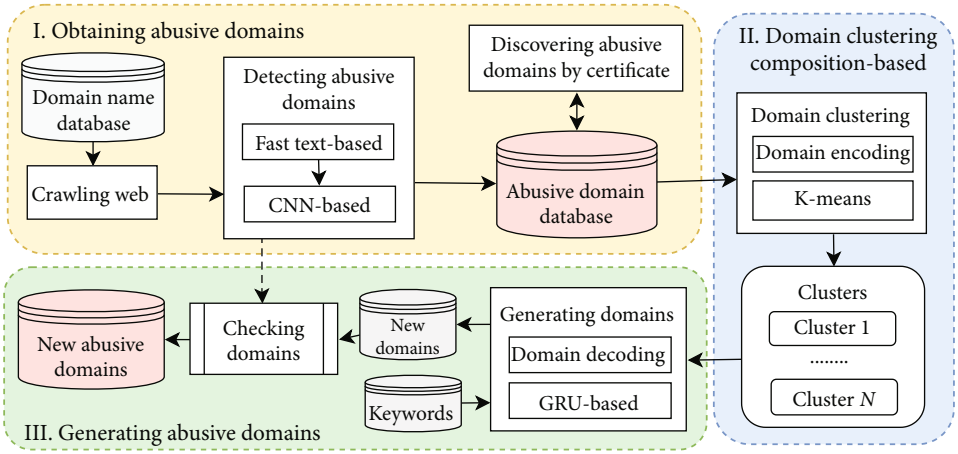


Figure 6: The process of generating new abusive domain names based on GRU-RNN.

We need to emphasize two points. On the one hand, in order to get as many websites with the same composition rules as possible, we get their web content in order based on the initial order of the domain names. On the other hand, we only need to obtain the web content of a small number of domains out of 260 million domains to meet our needs.

3.1.2. Detecting Abusive Domains Based on fastText. As introduced in Section 2, many previous studies detected the web page source codes of domains to determine whether they are pornographic or gambling. Also, both acquiring web page source code and abusive domain detection based on text are faster than acquiring web page snapshots and abuse detection based on the images. Therefore, text-based detection is the optimal solution when a large number of domains need to be detected while ensuring high efficiency and accuracy.

The fastText [29] is a natural language processing (NLP) library generally used for text representations and classification. The fastText does not need to rely on feature engineering like machine learning models for classification, and the classification effect does not depend on the selection of effective features. At the same time, although text classification based on deep learning can achieve good results and does not require feature engineering, the training speed is slow and the training conditions are high, so it cannot be used in large-scale text classification tasks. Therefore, the fastText model is widely used in text-based classification tasks because of its fast speed and good effect. Finally, this paper builds an abusive domain classifier with fastText based on HTML source codes. The process of detecting pornography and gambling domain names based on text is shown in Figure 7.

(1) Training and test sets. The training and test sets contain text samples that have been labeled as abusive or benign types. The text comes from the requests-based web crawler that gets the HTML source code of a large number of domains and extracts the key HTML tags content, i.e., title, description, and keywords. On the one hand, we obtain the source code of websites with a high traffic ranking in China from Alexa [30], and these text messages are labeled as benign. On the other hand, we obtain the source code of the web pages of the domains provided by Domain Monitor introduced above and filter out the text content matching pornographic and gambling keywords. These keywords (shared in GitHub [31]) are the more frequent Chinese words in pornography and gambling websites that we collected manually in the early stages, such as 做爱 (sex), 成人电影 (adult movies), and 澳门娱乐场 (Macau casinos). In addition, for the initially obtained benign and abusive texts, we manually filtered them again to ensure that the texts were indeed pornographic or gambling. In the end, we get a total of 31,667 benign and 177,963 abusive texts as training and test sets, which we will describe in detail in Section 4.1

(2) Text preprocessing. The task of text preprocessing for our dataset takes a few steps to convert the data into a convenient form that we can feed into the fastText classifier. First, since Chinese sentences are not separated by spaces like English sentences, we use the open-source tool Jieba [32] to split Chinese sentences into words. Jieba Chinese text segmentation is the best Python Chinese word segmentation module, and a lot of research relies on its excellent results. For example, the pornographic Chinese text "亚洲成人片不卡无码⊠天看片免费高清观看⊠国内自拍视频在线" (watch Asian adult movies and selfie porn videos for free every day) is divided into "亚洲 (Asian) 成人片 (adult videos) 不卡 (fluency) 无码 (codeless) 天天 (every day) 看片 (AV) 免费 (free) 高清 (high definition) 观看 (watch) 国内 (domestic) 自拍 (selfie) 视频 (videos) 在线 (online)." Second, in this step, we remove the repeated words after the sentence is divided. Also, remove meaningless words or symbols from the set of words, including stop words and special symbols, like #, ∗, and &, etc.

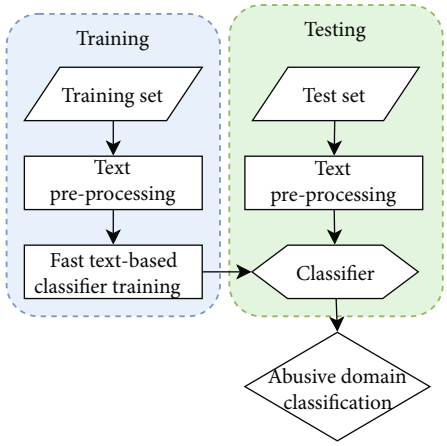


Figure 7: The process of detecting pornography and gambling domains based on fastText.

Finally, the fastText requires the labeling of each line in the database to be in a particular format like this: "__label__<label_name> <text>." For example, for pornographic text, it is "__label__ abusive 亚洲成人片不卡无码天天看片免费高清观看国内自拍视频在线." Another example, of benign text, is "__label__benign 百度一下⊠你就知道."

(3) FastText-based classifier training and classification. Text classification mainly uses a classifier to label the text of an unknown category, so the most important part of classification is the selection of the classification algorithm. The classification task proposed in this paper uses the fastText library by Facebook. Since the method needs the classification of texts with our prelabeled dataset, we used the supervised technique for text classification

Overall, the detection model we built based on fastText can detect a large number of domains containing pornographic and gambling texts. The detection model has the advantages of high accuracy, low resource consumption, and high efficiency, as detailed in Section 4.1.

3.1.3. Detecting Abusive Domains Based on CNN. As explained in Section 2.1.2, while some of the site's HTML information (title, description, and keywords) are related to pornography or gambling, the actual content presented on the site does not. As a result, we need to conduct additional research to determine whether the website is abusive. This section investigates the algorithm for detecting abusive domain names based on web page snapshots. Most pornographic or gambling websites have significantly different front-end design styles from benign websites. Therefore, features of both pages are automatically extracted by CNN, which are used to detect whether a domain name is being abused or not.

Abusive domain name detection based on web snapshots can be tried as an image binary classification problem, i.e., domain name snapshots are classified into benign and abusive results. In this paper, CNN is used to train the image recognition model, and the schematic diagram of the convolutional neural network structure is shown in Figure 8.

We input the web page snapshot into the convolutional neural network after it has been converted into an RGB 3D tensor of size 1600∗1000∗3. ReLU is applied nonlinearly to the output of the convolutional layer and the penultimate fully connected layer. The last fully connected layer uses a Sigmoid function to map the output to between 0 and 1 to obtain the probability that the input snapshot is legal for binary classification. In the training of the model, the loss function is a binary cross-entropy loss function, as shown in Equation (1).

$$\text{Loss} = \frac{-\sum_{i=0}^{n}(y_i \log(f_i(x_i))) + (1-y_i)\log(1-f_i(x_i))}{n}, \tag{1}$$

where n denotes the total number of output nodes, y_i is the real label corresponding to the i_{th} category, and $f_i(x)$ is the output value of the corresponding model.

Due to the large resolution of the input images, we increase the depth and width of the network in order to allow the neural network to extract its deep abstract features, reduce the number of parameters, and improve the classification accuracy. That is, the convolutional kernel width is 3 for each convolutional layer, and the convolutional kernel width is 2 for the pooling layer.

In order to improve the detection efficiency and accuracy of detecting pornographic and gambling domains, we combine two methods based on text filtering and image-based detection. That is, we first get a large number of suspected pornography and gambling websites by text-based filtering methods, then get snapshots of these websites, and finally detect whether these websites are really pornography and gambling websites by image-based methods.

3.1.4. Discovering Abusive Domains by Certificate. As described in Section 2.1.1, many abusive domains' certificates contain other abusive domains with a fairly similar character composition that can share these certificates. As a result, we developed tools to extract abusive domains from the certificates. Additionally, we extracted the primary domain name portion of the fully qualified domain names (FQDNs). For instance, for the domain https://www.abusive-domain.com, abusive-domain.com is the portion on which we concentrate our efforts. This enables us to acquire the maximum number of domain name composition rules feasible.

3.2. Domain Clustering Based on Composition. As described above, there are character compositional similarities between the different domain names in the set of abusive domain names. In other words, there are many different malicious domain names, but they come from various forms of character composition. In order to distinguish the abusive domain names belonging to different composition forms, we first cluster the abusive domain names based on the composition similarity of domain characters in order to provide training data for the domain name generation model.

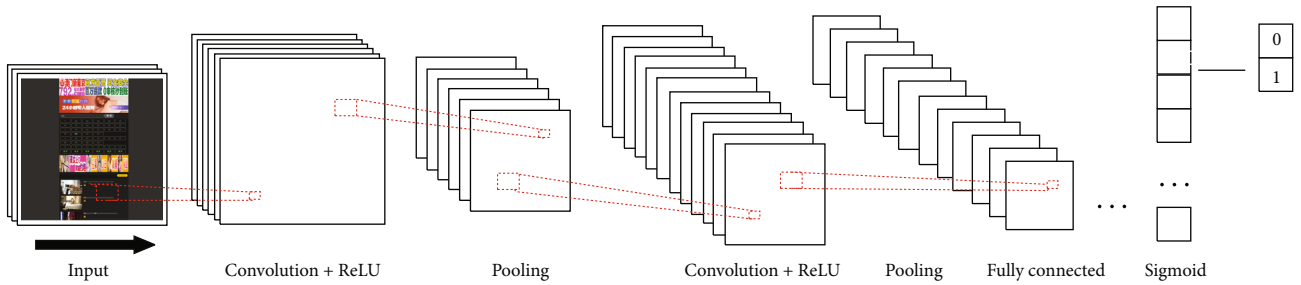


Figure 8: The process of identifying abusive website snapshots using convolutional neural networks.

3.2.1. Domain Encoding and Decoding. In order to achieve similar domain name clustering and interconversion with neural network acceptable input and output forms, encoding and decoding rules for domain names are constructed based on domain name composition features.

(i) *Domain Encoding*. Domain names are formed by the rules and procedures of the Domain Name System (DNS). The first-level set of domain names is the top-level domains (TLDs), including the generic top-level domains (gTLDs), such as .COM, .ORG, and .NET, and the country code top-level domains (ccTLDs). Below these top-level domains in the DNS hierarchy are the second-level and third-level domains (2LD and 3LD) that are typically open for reservation by end-users who wish to connect local area networks to the Internet, creating another publicly accessible Internet resource, or run websites. For example, for the domain name http://google.com, the top-level domain is .COM and the second-level domain is Google. We divide the domain name into two parts, i.e., 2LD.TLD, and encode the top-level domain and the second-level domain of the domain name, respectively. The second-level domains are all composed of letters a-z, numbers 0-9, and the ligature "-," while the top-level domains as a whole act as a specific unit due to their nondetachable nature. Therefore, we number all top-level domains and individual characters in the abusive domain dataset to form a domain-numbered dictionary. When encoding a domain name, first, we convert all characters and top-level domains in the domain name to their corresponding numbers, forming a domain-numbering vector of variable length. Then, the domain number vector is filled to the specified length with null characters to obtain a domain number vector of definite length composed of numbers.

(ii) *Domain Decoding*. Domain name decoding is the process of converting the domain name number vector output from the neural network into a domain name character vector. The domain name is obtained by looking up the corresponding characters and top-level domain names according to the character numbers in the domain character dictionary, obtaining a domain name character vector of definite length, and then removing the trailing null characters

3.2.2. Abusive Domains Clustering. In this paper, we use the K-means algorithm [33] to cluster domain names and divide the abusive domain names with similar character compositions into the same cluster set. The basic idea of the K-means algorithm is as follows.

(i) K domain feature vectors are selected from the dataset as the clustering centers

(ii) For each other domain feature vector, calculate its Euclidean distance [34] from all the cluster centers and assign it to the cluster center with the closest distance

(iii) Update the cluster centers of all cluster sets to the mean value of all domain feature vectors in the cluster set and calculate the squared distance sum $J(C)$ (as shown in Equation (2)) values of all samples to their category cluster centers

(iv) Finally, determine whether the clustering center and $J(C)$ value have changed; if they have, return to the second step to continue the iteration, and vice versa, end the algorithm

$$J(C) = \sum_{k=1}^{K}\sum_{i=1}^{n} d_{ki}\|x_i - y_k\|^2,$$
$$\text{where } d_{ki} = \begin{cases} 1, \text{if } x_i \in c_k \\ 0, \text{if } x_i \notin c_k \end{cases}, \tag{2}$$

where C is the set of clusters, K is the number of clusters, n is the number of sample data, x_i is the data point, and y_k denotes the cluster center of the k_{th} cluster set. Finally, we aggregate the abusive domain names with similar character compositions into K clusters. After that, new abusive domain names are generated based on each cluster.

3.3. Generating New Abusive Domains. The new abusive domain name generation problem can be viewed as a character sequence prediction problem. As described in Section 2.2, LSTM is a special kind of recurrent neural network that has been successful in dealing with machine translation and

sequence problems. However, the algorithm is slow to converge because of its large number of parameters. Therefore, in this paper, GRU is used to build the neural network, which works on the same principle as the LSTM layer but with computational simplifications making the operation less expensive, while the difference in model performance is not significant.

3.3.1. Generating Domains Based on GRU

(1) Building Generation Model

The schematic diagram of the structure of the GRU-based malicious domain name generation neural network is shown in Figure 9.

The first embedding layer accepts an integer domain number vector of definite length as input and will output a meaningful embedding vector of definite length. Each component of the vector consists of floating-point numbers, which can describe the relationship between the characters of the domain name in a specific way. The second layer of the GRU layer accepts the feature vector input of the previous layer and outputs the information of the character of the domain name at the next moment. The third layer of the fully connected layer acts as a classifier, which converts the input of the second layer into a vector of the size of the domain-numbered dictionary and outputs it. Finally, the output vector of the third layer is converted into a logarithmic probability distribution by LogSoftMax and output.

The steps of generating a batch of domain name vectors based on the GRU-based malicious domain name generation neural network are as follows:

(1) First calling the pseudo-random generator to select a batch of first character numbers from the number set corresponding to the set of letters and numbers and forming the corresponding domain number vector, that is, a tensor of 1∗batch-size, and input to the embedding layer, where batch-size is customized by the user during the generation process

(2) The embedding layer converts each input character number into an embedding-dim dimensional embedding vector to obtain the embedding feature in the shape of 1∗batch-size∗embedding-dim and outputs it to the GRU layer. During this experiment, embedding-dim is 128

(3) The GRU layer converts the input of embedding features into a tensor in the shape of 1∗batch-size∗hidden-dim, which contains the information to predict the character number of the next batch, and inputs it to the fully connected layer. Hidden-dim is the size of the hidden layer of the GRU layer. During the experiment, hidden-dim is 256

(4) The fully connected layer converts the input tensor into a tensor in the shape of batch-size∗char-count and outputs it. LogSoftMax converts each line to a probability distribution of the next character number to be generated, where char-count is the size of the domain-numbered dictionary. During this experiment, char-count is 627

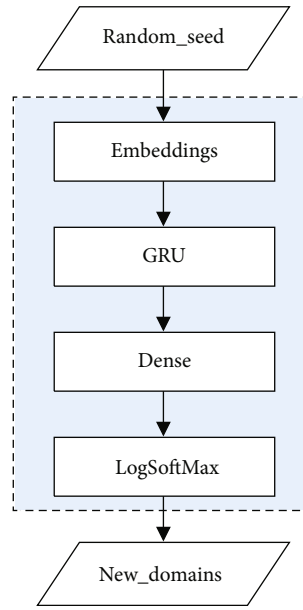


Figure 9: The structure of the GRU-based malicious domain name generation neural network.

(5) For each row, the character number with the highest probability is selected as the number of the next batch of characters generated. The character number of the next batch is used as the new input to the neural network

(6) Repeat steps (2) to (5) until the length of the domain name reaches the maximum length or the generated character number belongs to the top-level domain character numbers, then stop generating. The initial input character number and the domain name number generated in each step form a domain name number vector in the generation order. The domain name number vector is decoded into the corresponding domain name characters to obtain a batch of generated domain name vectors

(2) Training Generation Model

The steps of training GRU-based malicious domain name generation neural networks are as follows:

(1) Read model configuration information and related parameters

(2) Based on the clustering results, all the original domain name data in one of the categories that have not yet participated in model training are read as the model training dataset

(3) The original domain name data is encoded and converted into a domain name number vector, and the length of the domain name number vector is filled

to max-length. During this experiment, max-length is 50

(4) Configure the Adam optimizer to adapt to gradient changes to adjust different learning rates according to parameter changes

(5) Get a batch of training data shaped as max-length∗ batch-size

(6) Each column of training data is a domain name number vector. The first element to the max-length-1 element of each domain name number vector is combined into an input domain vector named input-vector. A batch of input-vector is combined into a tensor in the shape of (max-length-1) ∗ batch-size. The second element to the max-length element of each domain name number vector is combined into a target vector named target-vector, and a batch of target-vector is combined into a target tensor in the shape of (max-length-1) ∗ batch-size

(7) The model is learned and fitted on the input tensor and the target tensor. The loss value between the prediction result of the model under the input tensor and the target tensor is calculated using cross-entropy as the loss function. The trainable parameters are updated by backpropagation. The network parameters are updated by the optimizer

(8) Repeat steps (5) to (7) until the maximum number of iterations is reached or the network loss value stabilizes, and save the network parameters when the loss value is below the set threshold during the training process

(9) Repeat steps (2) to (8) until the original domain name data of all categories in the clustering results are used as the model training dataset to participate in the neural network training

3.3.2. Checking Domains. In this step, we first check if the generated domains are configured with IP addresses. For domains with configured IP addresses, we then try to obtain the web content of these domains. Finally, using the method, we devised (described in Section 3.1) to detect whether these domains are abusive or not.

3.4. Evaluation Metrics. In this paper, we use standard accuracy (*Acc*, Equation (3)), precision (*P*, Equation (4)), recall (*R*, Equation (5)), and F1-score (*F*1, Equation (6)) as the classification evaluation metrics to evaluate the performance of abusive domain name detection. The specific formulas are as follows:

$$Acc = \frac{TP + TN}{TP + FP + TN + FN}, \tag{3}$$

$$P = \frac{TP}{TP + FP}, \tag{4}$$

$$R = \frac{TP}{TP + FN}, \tag{5}$$

$$F1 = \frac{2 * P * R}{P + R}, \tag{6}$$

where *TP* is true positives, *TN* is true negatives, *FN* is false negatives, and *FP* is false positives, respectively. We define the abusive domain names as positive and the benign ones as negative.

As we described above, pornographic and gambling domains detected using the fastText-based classifier are again detected using the CNN-based classifier. Therefore, we use Equation (7) to evaluate its joint precision (P_{joint}).

$$P_{\text{joint}} = 1 - (1 - P_{\text{text}}) * \left(1 - P_{\text{image}}\right), \tag{7}$$

where P_{text} denotes the precision of detecting abusive domains based on text, and P_{image} denotes the precision of detecting abusive domains based on the image.

4. Experimental Results

In this section, we mainly show the abusive domain detection performance and the results of the generated abusive domains.

4.1. Abusive Domain Database

4.1.1. Performance of fastText-Based Detection. In order to evaluate the performance of the fastText-based detection models, we used a 10-fold cross-validation strategy over the dataset. First, the dataset is split into 10 folds, then the seven folds are trained, and the remaining one is used for testing. This process is repeated ten times. Finally, all the metrics on the validation folds are averaged and a better estimate of the performance is achieved. Based on the method introduced in Section 3.1, we obtained the text information of benign and abusive domain names as shown in Table 1.

We trained and tested the model on Apple's Mac mini M1 version (8 CPU cores and 16 GB RAM). Under our normal conditions of using the device, such as opening the browser, PyCharm software, the model training took only 4.3 seconds. In addition, the model took only 1.2 seconds to complete the classification of 62,751 domain names. The values of *P*, *R*, *F*1, and *Acc* of the model in detecting abusive domain names are 0.98, 0.97, 0.98, and 0.96, respectively. Thus, the abusive domain name detection model base on fastText we built has very good performance with high efficiency and low resource consumption.

4.1.2. Performance of CNN-Based Detection. Based on text detection of pornography and gambling domains, we filtered out a large number of pornography and gambling websites. We used the selenium-based crawler to obtain a total of 37,266 web snapshots of pornography and gambling domains. In addition, we obtained 27,132 web snapshots of domains provided by Alexa as benign samples. In the experiment, this paper divides the original domain snapshots into

Table 1: The summary of the training and test datasets for fastText-based detection.

Category	Training	Test
Abusive domains	124,686	53,277
Benign domains	22,193	9,474
Total	146,879	62,751

Table 2: The summary of the training, verification, and test datasets for CNN-based detection.

Category	Training	Verification	Test
Abusive domains	26,085	3,725	7,456
Benign domains	18,992	2,713	5,427
Total	45,077	6,438	12,883

training set, validation set, and test set according to the 7 : 1 : 2. The dataset division is shown in Table 2.

The GPU graphics card used in this experiment is the Quadro GV100 with 32 GB of video memory. TensorFlow (https://www.tensorflow.org, accessed on 6 May 2022), Matplotlib, Keras (https://keras.io, accessed on 6 May 2022), and other services are configured in the operating system (OS) system Ubuntu 20.04.3LTS environment. In this paper, the GPU-accelerated convolutional neural network was built with the Keras framework as the core. The evaluation metrics of model classification include training accuracy, training loss, validation accuracy, and validation loss, where validation accuracy and loss are used to determine whether the model is overfitted during the training iteration and to evaluate the generalization ability of the model. The training accuracy and loss values are used to evaluate the model's performance on the training set. The variation of the classification accuracy and loss values of the convolutional neural network with the number of training epochs during the model training are shown in Figure 10.

As shown in Figure 10, the model's classification accuracy went from 64% to 92% after 26 epochs of training. The accuracy increases rapidly in the first 10 epochs, fluctuates slightly between 11 and 26 rounds, and starts to grow slowly after 26 epochs. This indicates that the training of the model has converged. The trend of the model training loss value in Figure 10 is basically opposite to the trend of the model training accuracy. The loss value of the model decreases from 6.6 to 2.6 after 26 epochs of training. The loss value of the model starts to fluctuate after 26 and basically tends to be constant.

The variation of the validation classification accuracy and the validation loss value of the convolutional neural network with rounds during the model training are shown in Figure 11.

The accuracy and loss trends in Figure 11 are generally consistent with Figure 10. The validation accuracy values in Figure 11 start to fluctuate when the rounds reach 26, indicating that the fit has converged. After 26 epochs of training, the final accuracy increases from 0.77 to 0.91. The validation loss value tends to level off after 26 epochs of training, with only a slight vibration, which indicates the strong generalization ability and high accuracy of the model.

Finally, the experimental results show that the model built in this paper has an accuracy of 0.95 on the training set, 0.90 on the validation set, and 0.91 on the test set. Based on the test set, the model's values of P, R, and $F1$ in detecting abusive domain names are 0.92, 0.93, and 0.92, respectively.

Furthermore, as we introduced in the above section, we use CNN-based identification in the set of pornographic or gambling domains discovered by the fastText-based model. That is, we use the built CNN-based model to filter out benign domains that are misclassified as gambling or pornography from the set of abusive domains in this step. After evaluation, the CNN-based model achieves an accuracy value of 0.98 in detecting benign domains in this abusive domain set. Based on Equation (7), the joint model built in this paper achieves an accuracy (P_{joint}) of 0.99 in detecting pornographic and gambling domains.

4.2. Generating New Abusive Domains

4.2.1. Domain Clustering. When using the K-means algorithm to cluster domain names that are similar in composition, the first step is how to determine the appropriate K value. The elbow method [35] is proposed to explain and verify the consistency of clustering analysis to assist in the determination of the optimal number of clusters in the dataset. The core idea of the elbow method is that if the value of K is much smaller than the optimal number of clusters, as the value of K increases, which will greatly increase the degree of aggregation of each cluster, so the value of $J(C)$ will decrease sharply. When the value of K increases close to the optimal number of clusters, the decrease of the $J(C)$ value slows down as the value of K increases. And when the value of K reaches the optimal number of clusters, continuing to increase the value of K will cause $J(C)$ to level off. Therefore, $J(C)$ decreases sharply and then flattens out as the K value increases, and the optimal K value is the K value at the inflection point.

Therefore, we determine the correct number of clusters according to the elbow method. Its calculation formula is shown in Equation (2). We selected different K values to cluster the abusive domain name dataset while calculating the $J(C)$ values, as shown in Figure 12. It can be seen that when the curve has an inflection point at K =5 or 6.

The silhouette coefficient or silhouette score [36] is a metric used to calculate the goodness of a clustering technique. The silhouette score ranges from -1 to 1. The negative value indicates that the sample is assigned to the wrong set of clusters, and the assignment is not satisfactory. When the value is positive, the larger the value, the smaller the distance between samples of the same category, and the larger between samples of different categories, the better the clustering effect. The expression of the sample silhouette coefficient s is shown in Equation (8).

$$s = \frac{b - a}{\max(a, b)}. \tag{8}$$

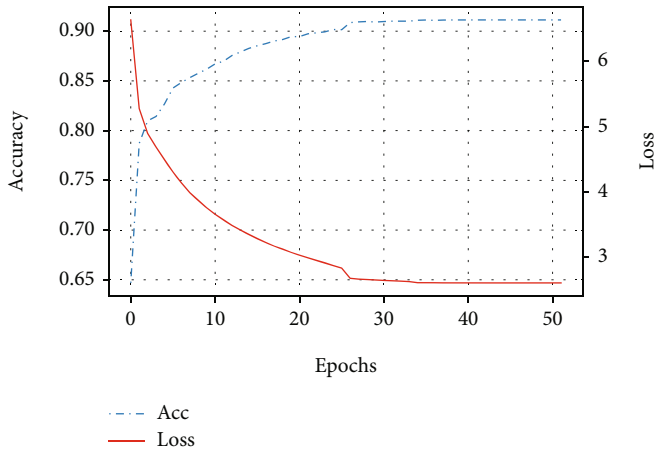


Figure 10: The accuracy and loss of model classification with epochs during model training.

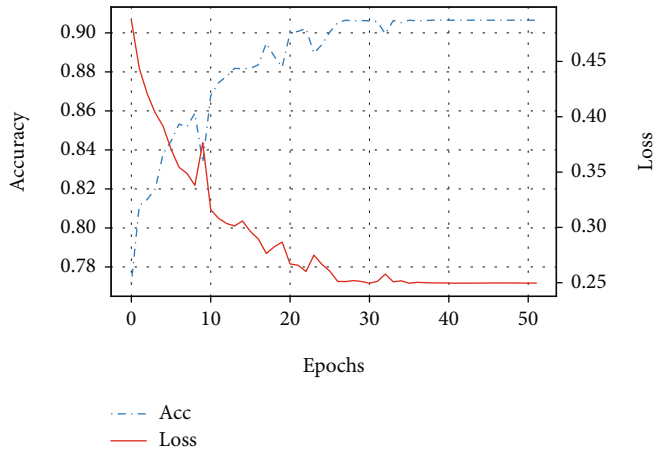


Figure 11: The accuracy and loss of model classification with epochs during model testing.

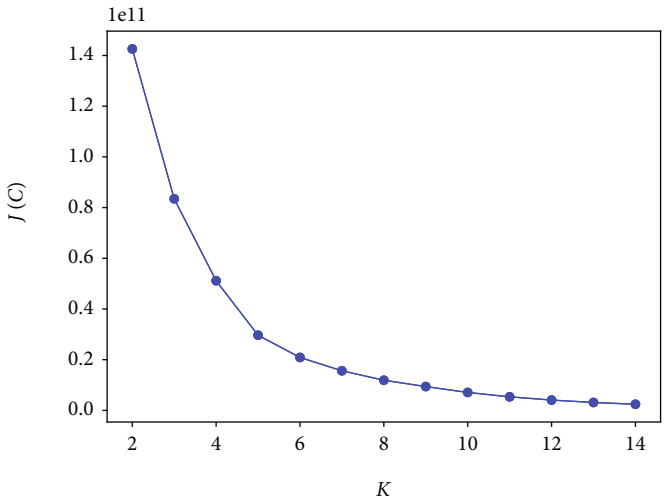


Figure 12: The decline curve of $J(C)$ as the K value increasing.

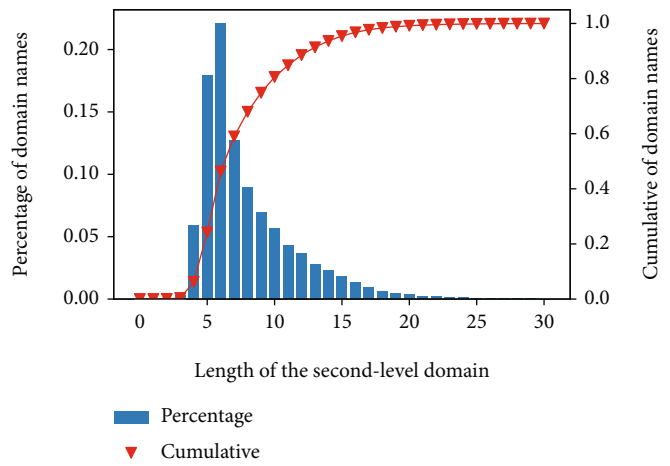


Figure 13: Distribution of second-level domain length.

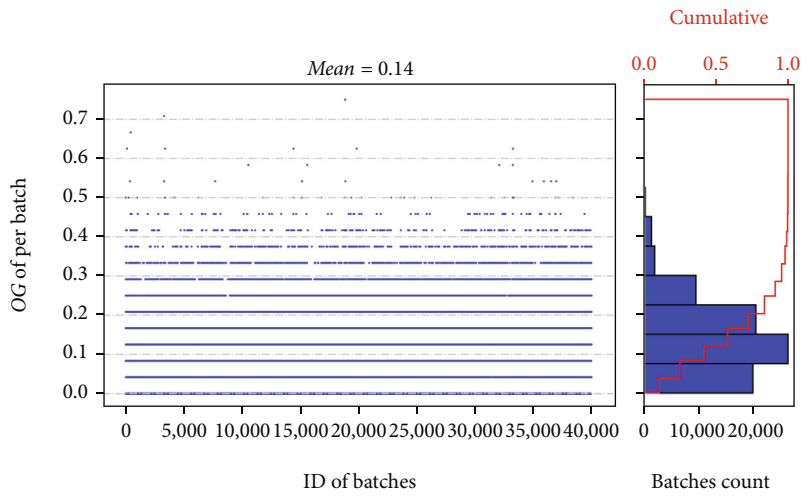


Figure 14: Distribution of new online domain names (2022-03-23).

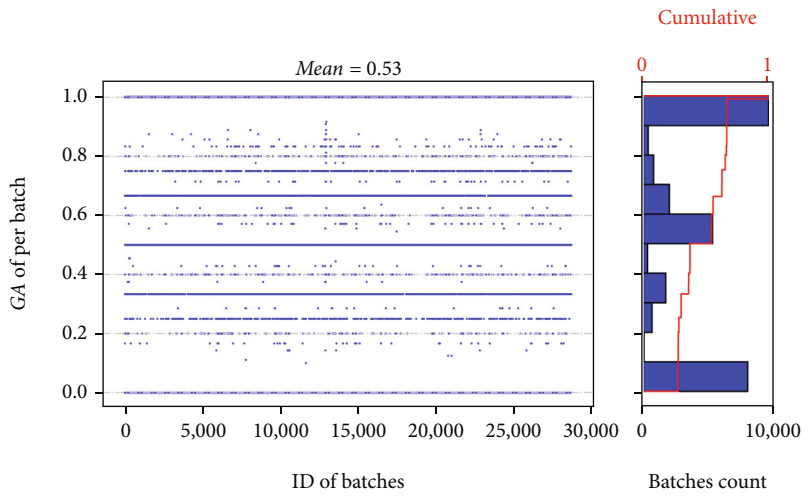


Figure 15: Distribution of new abusive domain names (2022-03-23).

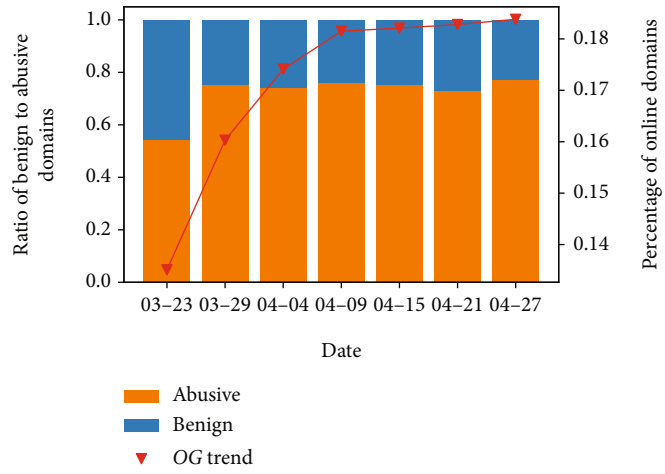


Figure 16: Increasing *GA* and *OG* values for new domain names generated over time.

In Equation (8), a is the average distance between the sample and other samples in the same cluster, and b is the minimum average distance between the sample and not in the same cluster samples.

The silhouette coefficient S of the sample set is the average of the silhouette coefficients of all samples, and its expression is as in Equation (9).

$$S = \frac{\sum_{i=1}^{n} s_i}{n}, \tag{9}$$

where n is the total number of samples, and s_i is the value of the i_{th} sample silhouette coefficient.

Finally, the values of the silhouette coefficients of the sample set are 0.866 and 0.831 when K =5 and 6, respectively. When K=5 indicates that the clustering effect is relatively satisfactory, the abusive domains within the uniform cluster sets are more similar. As a result, we divided the abusive domain names into five categories separately to generate new abusive domain names.

4.2.2. Generating Domain Names. The goal of abusive domain name generation is to generate domains that are as online (with web content) and as abusive as possible. We measure the performance of the generation model in terms of two metrics: the online rate of generating domain names (*OG*, as shown in Equation (10)) and the generation rate of abusive domain names (*GA*, as shown in Equation (11)).

$$OG = \frac{OGD}{GD}, \tag{10}$$

$$GA = \frac{AOD}{OGD}. \tag{11}$$

OG indicates the percentage of domains with web content to these generating domains. *GA* indicates the percentage of online domains that are pornographic or gambling domains. *GD* refers to the total number of generated domains. *OGD* refers to the number of online generated domains. *AOD* is the number of online domains that are abusive.

The neural network in the domain name generation model is built based on PyTorch (https://pytorch.org, accessed on 6 May 2022), and the model is configured and generated as described in Section 3.3. During the training process, the neural network weight parameters with model loss values below 0.3 are saved. In addition, for the abusive domains that have been divided into 5 clusters, 10 neural network weights are saved under each cluster, i.e., we end up with a total of 50 neural network weights.

Then, the keywords that compose the domain names are fed to the trained neural network model, and the model can generate the domain names with the maximum probability of abuse based on those keywords. Therefore, based on the configuration of the domain name generation model, 50 domain names can be generated for each keyword (or called batch), and the same batch does not contain the same domain name.

In the experiments, we determine the length of input keywords based on the length of the domainâ€™s second-level domain (e.g., the domain http://google.com, whose second-level domain is Google). As shown in Figure 13, which shows the distribution of the second-level domain length of benign and abusive domains we collected, it can be seen that more than 95% of the domains have a second-level domain length of less than 15. Therefore, we input 40,000 random strings (batches) (26 letters and ten digits) with a length less than 15 into the domain generation model. Finally, we got a total of 964,112 new domain names.

We use the requests-based crawler (introduced in Section 3.1) to crawl the web content of the domains in order to check whether the generated domains are online or not. The percentage of online domains in each batch of 50 generated domains is shown in Figure 14. We can discover that the average *OG* value is about 0.14, i.e., about 130,220 of the generated 1 million domains are with web content. In addition, the maximum and minimum values of *OG* for the batches of generated domains are 0.68 and 0.06, respectively. We can also find from the cumulative graph that the *OG* for each batch is concentrated between 0.06 and 0.3. This indicates that the number of online domains is related to the keywords entered into the generation model. By

selecting appropriate keywords, the number of online domains can be enhanced.

Next, we analyze how many of the generated online domains are malicious. Figure 15 shows the distribution of the percentage of generated online domains with pornographic or gambling websites in each batch. The average *GA* value was 0.53 on March 23, 2022, which means that about 70,318 domains were pornographic or gambling. Thus, this suggests that the domain name generation model we built can discover a large number of new pornographic and gambling domains in order to expand the list of abusive domains.

In addition, during our detection of all newly generated domains for more than one month, we found that more and more new domain names were gradually coming online, and most of the online domains were pornographic or gambling domains, as shown in Figure 16. The *OG* value of generated new domains improved from 0.14 to 0.18, i.e., about 48,205 domains came online in a month. And the percentage of new online domains that are abusive increased from 0.54 to 0.78, which indicates that a large number of pornographic or gambling domains came online over time and then spread malicious information. The experiments show that the domain generation model used in this paper can find a lot of pornographic and gambling domains in advance.

To summarize, we entered 40,000 keywords into the domain generation model and generated a total of 964,112 unique domain names. Eighteen percent of these domains (177,217 domains) serve web pages, with 127,596 domains serving pornographic or gambling sites. It turns out that with this domain name generation model, we can get a lot of new pornographic and gambling domain names with the same composition as the old domains.

5. Conclusion

The first step in blocking and dealing with pornographic and gambling domains is to discover them. The more quickly, precisely, and early these abusive domains are handled, the more harm they cause to people, particularly children and minors, can be mitigated. In this paper, we developed a two-layer detection system to quickly and precisely detect pornography and gambling domains using fastText and CNN models. In particular, in order to discover more abusive domains earlier, we proposed a domain generation model based on GRU for rapidly generating new abusive domain names from known ones. The experimental results demonstrate that our domain name detection and generation model is capable of discovering a large number of pornographic and gambling domains.

Moreover, it should be noted that the number of new gambling and pornographic domains that can be generated is more related to the sample of already existing pornographic and gambling domains. The larger the number of domains in the sample, and the more domains with similar character composition, the better the generated domains. Therefore, the limitation of this paper mainly comes from the number and quality of the sample domain names.

In the future, first, we should detect DGA domains using the idea of text-based generation and find domains generated using the same algorithm. Second, we should study the relationship between different keywords (length and composition) and the generated domain names that are more likely to be abusive. Finally, we would like to apply the detection techniques of this paper for pornography and gambling domains to discover phishing attack activities. Also, we would like to discover potential attacks by generating domain names that are similar to the target domain (e.g., http://google.com).

Abbreviations

CNN: Convolutional neural network
DGA: Domain generation algorithm
LSTM: Long-Short-Term Memory
GRU: Gated recurrent units
NL: Natural language processing
FQDN: Fully qualified domain name
DNS: Domain name system
TLD: Top-level domain
TP: True positive
TN: True negative
FN: False negative
FP: False positive
OG: Online rate of generating domains
GA: Generation rate of abusive domains
RNN: Recurrent neural network.

Data Availability

https://reurl.cc/0p27db, access password: nist@HIT.

Conflicts of Interest

The authors declare no conflict of interest.

Authors' Contributions

Y.C. and Z.Z. contributed to the conceptualization; Y.C. and H.J. contributed to the methodology; Y.C. and H.J. were responsible for the software; Y.C. and H.J. contributed to the validation; T.C. contributed to the formal analysis; Y.C. contributed to the investigation; Y.C. was responsible for the resources; T.C. contributed to the data curation; Y.C. and H.J. contributed to the writing—original draft preparation; Y.C. and T.C. contributed to the writing—review and editing; Y.C. and T.C. contributed to the visualization; Z.Z. and Y.D. contributed to the project administration. All authors have read and agreed to the published version of the manuscript.

Acknowledgments

This research was funded by the Natural Science Foundation of Shandong Province [Grant No. ZR2020KF009] and the Young Teacher Development Fund of Harbin Institute of Technology [Grant No. IDGA10002081].

References

[1] H. Yang, K. Du, and Y. Zhang, "Casino royale: a deep exploration of illegal online gambling," in *Proceedings of the 35th Annual Computer Security Applications Conference*, pp. 500–513, San Juan, PR, USA, December 2019.

[2] D. C. Hodgins and R. M. Stevens, "The impact of COVID-19 on gambling and gambling disorder: emerging data," *Current Opinion in Psychiatry*, vol. 34, no. 4, pp. 332–343, 2021.

[3] A. Hakansson, F. Fernandez-Aranda, and J. M. Menchon, "Gambling during the COVID-19 crisis – a cause for concern," *Journal of Addiction Medicine*, vol. 14, no. 4, pp. e10–e12, 2020.

[4] "Pornography is booming during the covid-19 lockdowns," May 2022, https://www.economist.com/international/2020/05/10/pornography-is-booming-during-the-covid-19-lockdowns.

[5] H. A. Awan, A. Aamir, M. N. Diwan et al., "Internet and pornography use during the COVID-19 pandemic: presumed impact and what can be done," *Frontiers in Psychiatry*, vol. 12, p. 220, 2021.

[6] V. Cerdan Martinez, D. Villa-Gracia, and N. Deza, "Pornhub searches during the Covid-19 pandemic," *Porn Studies*, vol. 8, no. 3, pp. 258–269, 2021.

[7] M. Brodeur, S. Audette-Chapdelaine, A. C. Savard, and S. Kairouz, "Gambling and the COVID-19 pandemic: a scoping review," *Progress in Neuro-Psychopharmacology and Biological Psychiatry*, vol. 111, article 110389, 2021.

[8] Y. Cheng, Y. Liu, L. Wang, Z. Zhang, T. Chai, and Y. Du, "Evaluating the effectiveness of handling abusive domain names by internet entities," *Electronics*, vol. 11, no. 8, p. 1172, 2022.

[9] R. Yang, X. Wang, and C. Chi, "Scalable detection of promotional website defacements in Black Hat SEO campaigns," in *Proceedings of the 30th USENIX Security Symposium (USENIX Security 21)*, pp. 3703–3720, Vancouver, B.C., Canada, 2021.

[10] M. Stamp, M. Alazab, and A. Shalaginov, *Malware Analysis Using Artificial Intelligence and Deep Learning*, Springer, Berlin/Heidelberg, Germany, 2021.

[11] P. Lison and V. Mavroeidis, "Automatic detection of malware-generated domains with recurrent neural models," 2017, https://arxiv.org/abs/1709.07102.

[12] R. R. Curtin, A. B. Gardner, and S. Grzonkowski, "Detecting DGA domains with recurrent neural networks and side information," in *Proceedings of the 14th International Conference on Availability, Reliability and Security*, pp. 1–10, Canterbury, CA, United Kingdom, 2019.

[13] C. Xu, J. Shen, and X. Du, "Detection method of domain names generated by DGAs based on semantic representation and deep neural network," *Computers & Security*, vol. 85, pp. 77–88, 2019.

[14] B. Bharathi and J. Bhuvana, "Domain name detection and classification using deep neural networks," in *Proceedings of the International Symposium on Security in Computing and Communication*, pp. 678–686, Bangalore, India, 2018.

[15] F. Ren, Z. Jiang, and J. Liu, "Integrating an attention mechanism and deep neural network for detection of DGA domain names," in *Proceedings of the 2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*, pp. 848–855, Portland, OR, USA, November 2019.

[16] Y. Chen, R. Zheng, A. Zhou, S. Liao, and L. Liu, "Automatic detection of pornographic and gambling websites based on visual and textual content using a decision mechanism," *Sensors*, vol. 20, no. 14, p. 3989, 2020.

[17] J. Zhao, M. Shao, H. Peng, H. Wang, B. Li, and X. Liu, "Porn2-Vec: a robust framework for detecting pornographic websites based on contrastive learning," *Knowledge-Based Systems*, vol. 228, article 107296, 2021.

[18] Q. Wang, T. Luo, and D. Wang, "Chinese song iambics generation with neural attention-based model," 2016, https://arxiv.org/abs/1604.06274.

[19] B. L. Sturm, J. F. Santos, and O. Ben-Tal, "Music transcription modelling and composition using deep learning," 2016, https://arxiv.org/abs/1604.08723.

[20] Y. Luo and Y. Huang, "Text steganography with high embedding rate: Using recurrent neural networks to generate chinese classic poetry," in *Proceedings of the 5th ACM workshop on information hiding and multimedia security*, pp. 99–104, New York, NY, USA, June 2017.

[21] R. Fu, Z. Zhang, and L. Li, "Using LSTM and GRU neural network methods for traffic flow prediction," in *Proceedings of the 2016 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC)*, pp. 324–328, Piscataway, NJ, USA, November 2016.

[22] Y. Liu, D. Liu, and J. Lv, "Deep poetry: a Chinese classical poetry generation system," in *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 13626-13627, New York, NY, USA, February 2020.

[23] A. Bartoli, A. De Lorenzo, and E. Medvet, ""' Best dinner ever!!!": automatic generation of restaurant reviews with LSTM-RNN," in *Proceedings of the 2016 IEEE/WIC/ACM International Conference on Web Intelligence (WI)*, pp. 721–724, Omaha, NE, USA, October 2016.

[24] Y. Cao, R. Ji, L. Ji, G. Lei, H. Wang, and X. Shao, "l^2-MPTCP: a learning-driven latency-aware multipath transport scheme for industrial Internet applications," *IEEE Transactions on Industrial Informatics*, p. 1, 2022.

[25] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[26] "Download list of all domains," May 2022, https://domains-monitor.com.

[27] "Requests: HTTP for humans," May 2022, https://docs.python-requests.org/en/latest.

[28] "Selenium," May 2022, https://www.selenium.dev.

[29] "fasttext Â·PyPI," May 2022, https://pypi.org/project/fasttext/.

[30] "Alexa ranking _ Website traffic worldwide ranking _ Chinese website ranking," May 2022, http://www.alexa.cn/.

[31] "reporting_abusive_domains/abusive_keywords.txt at main Â·mrcheng0910/reporting_abusive_domains," May 2022, https://github.com/mrcheng0910/reporting_abusive_domains/blob/main/abusive_keywords.txt.

[32] "fxsjy/jieba: Jieba Chinese word segmentation," May 2022, https://github.com/fxsjy/jieba.

[33] A. K. Jain, "Data clustering: 50 years beyond K-means," *Pattern recognition letters*, vol. 31, no. 8, pp. 651–666, 2010.

[34] "Euclidean distance - Wikipedia," May 2022, https://en.wikipedia.org/wiki/Euclidean_distance.

[35] F. Liu and Y. Deng, "Determine the number of unknown targets in Open World based on Elbow Method," *IEEE Transactions on Fuzzy Systems*, vol. 29, no. 5, pp. 986–995, 2021.

[36] P. Berkhin, "A survey of clustering data mining techniques," in *Grouping Multidimensional Data*, pp. 25–71, Springer, Berlin, Heidelberg, Germany, 2006.

Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 1243041, 12 pages
https://doi.org/10.1155/2022/1243041

Research Article

A Sparse Feature Matching Model Using a Transformer towards Large-View Indoor Visual Localization

Ning Li,[1,2] Weiping Tu,[1,2] and Haojun Ai[3]

[1]*National Engineering Research Center for Multimedia Software, School of Computer Science, Wuhan University, Wuhan 430072, China*
[2]*Hubei Key Laboratory of Multimedia and Network Communication Engineering, Wuhan University, Wuhan 430072, China*
[3]*School of Cyber Science and Engineering, Wuhan University, Hubei 430072, China*

Correspondence should be addressed to Weiping Tu; tuweiping@whu.edu.cn and Haojun Ai; aihj@whu.edu.cn

Received 12 May 2022; Accepted 21 June 2022; Published 4 July 2022

Academic Editor: Yuanlong Cao

Accurate indoor visual localization has been a challenging task under large-view scenes with wide baselines and weak texture images, where it is difficult to accomplish accurate image matching. To address the problem of sparse image features mismatching, we develop a coarse-to-fine feature matching model using a transformer, termed MSFA-T, which assigns the corresponding semantic labels to image features for an incipient coarse matching. To avoid the anomalous scoring of sparse feature interrelationship in the attention assigning phase, we propose a multiscale forward attention mechanism that decomposes the similarity-based features to learn the specificity of sparse features, the influence of position-independence on sparse features is reduced and the performance of the fine image matching in visual localization is effectively improved. We conduct extensive experiments on the challenging datasets; the results show that our model achieves image matching with an average 79.8% probability of the area under the cumulative curve of the corner point error, which outperforms the related state-of-the-art algorithms by an improvement of 13% probability at 1 m accuracy for the image-based visual localization in large view scenes.

1. Introduction

Obtaining an accurate indoor location is a key for location-based services such as seamless indoor-outdoor integrated navigation and multimedia information push in smart cities and augmented/virtual reality applications [1]. The demand for the high-precision location-based services in large indoor spaces is also becoming increasingly urgent.

Visual indoor localization is currently the mainstream solution under the premise of high precision location [2]. The localization accuracy estimated with visual information exceeds that with wireless radio frequency (RF) signals, IMU, and geomagnetic signals. The RF signal is affected by multipath effects and signal fading, while IMU suffers from error accumulation, and they cannot compete with robust visual localization. Visual localization, i.e., estimating the camera pose by query image matching to the scene model, is a core problem under a large-view condition in computer vision. In the absolute pose estimation of a camera, it is necessary to estimate the pose in an indoor coordinate system using the information provided by the image database and 3D point clouds. The main challenge for the image-based [3–5] or structure-based [6, 7] indoor visual localization methods is to obtain the exact image feature matching (i.e., find the feature points corresponding to the query image from the candidate images) and complete the homography constraint in optimal camera pose estimation [8, 9]. However, in complex indoor scenes, especially images in large-view scenes with long viewing distances and wide baselines, which contain the visual information with sparse features, feature distortion or partially occluded makes it difficult to accomplish accurate feature matching of the visual localization. Similarly, some viewpoint changes in a wide range of viewing conditions lead to acute perspective distortion, which results in a little scene structural overlap between the query and the candidate images. Because image matching

focuses on the small part of an image [10], the variability of the scale and rotation of local features makes feature matching in large view scenes highly ambiguous and unable to accomplish accurate visual localization.

The precise correspondence of image features between the query and the database is a key to visual localization under a wide range of viewing conditions. The accuracy of image matching in such scenes can be improved by visual semantic information and spatial context [11]. The works [12, 13] extracted scene semantic information for consistency matching and used the geometric and semantic understanding of the scene to learn the new generative descriptors for positioning under failed scenes. These methods are able to eliminate the influences of illumination and occlusion for visual long-time localization. However, the accuracy of geometric descriptors [14] and semantic segmentation models [15] needs to be further improved for getting accurate geometric features and semantic annotation of the large-view indoor scene 3D model. For the image sparse feature matching of visual localization in large view scenes, the attention-based matching algorithm provides a promising approach [16, 17], the translational and rotational invariance of features is learned to enhance the expression of sparse features, and the different matching strategies are accomplished through different attention weights assignment, which can solve the ambiguous problem of feature matching in a large view scene with crossviewpoint. In the existing methods of self-attention weights and crossattention weights [17–19], the anomalous attention weights of the sparse feature points under weak texture scenes are prone to occur for location-independent feature points (i.e., feature points prone to distortion) because they are not subject to any constraints, leading to the pervasive weak texture image matching errors in large view and increasing visual localization errors.

To tackle the above challenges, we investigate the problems of ambiguous matching of sparse features and anomaly weights for visual feature correlation under large view scenes. We develop a coarse-to-fine feature matching model to remove the dependence on appearance-based reliable feature matching and reduce the effects of the large view and viewpoint changes. As shown in Figure 1, a key insight of our method is to learn the self-correlation among the image sparse features and crosscorrelation among the features on different image patches through semantic correlation and forward multiscale attention mechanism, which reduces the influence of image distortion and improves the matching accuracy of sparse feature points under a wide range of viewing conditions. The key contributions are summarized as follows:

(1) We develop a novel coarse to fine feature matching network with a transformer, termed MFSA-T, which solves the problem of sparse feature matching in large view scenes. Meanwhile, semantic match consistency and position correlation are exploited to improve the robustness of the refined matching model

(2) We propose a multiscale forward attention mechanism to solve the anomaly score of sparse feature point interrelationship and the attention weight on different image patches. This mechanism enables our network to decompose the similarity features to learn the specificity, which improves the matching accuracy of the sparse features in weak texture regions and refines the visual localization in large view scenes

(3) We achieve an average correct matching rate of 79.8% in large view scenes and reduce the localization error by 9.5% in wide baseline scenes of the public datasets, which outperforms the state-of-the-art image matching algorithms. The performance of image-based visual localization algorithms using the MFSA-T model in large-view scenarios is successfully improved

The rest of the paper is organized as follows. Section 2 discusses the existing studies related to this research and Section 3 illustrates the method regarding the developed sparse feature matching network in large views scenes. Finally, the experiment results along with their analysis and the summarization of the developments are discussed in Section 4 and Section 5, respectively.

2. Related Work

Robust visual localization in large view scenes is an essential problem in computer vision. The solution of this problem in difficult situations is not only a challenging task but also highly relevant in practice, such as augmented reality, multimedia information push, and autonomous robots. Large view scenes with extreme viewpoint changes, a wide baseline of view, and weak textures lead to acute perspective distortion and frequently bring on the few common matching parts between the query and the database. These challenges in visual localization attract a large number of researchers to investigate different visual problems [20]. In this section, we review and summarize the research on issues related to visual localization in a large view scene.

2.1. Feature-Based Localization. The mainstream visual localization algorithms for large-scale complex indoor scenes use local feature matching of the query image with the 3D model from the structure from motion (SFM) [21] of the scene, such as SIFT [22] and FREAK [23]; the homography matrix formed by the corresponding features after RANSAC filtering is solved by perspective-n-point (PnP) [24] to estimate the pose of the query image [9]. To eliminate the influence of viewpoint changes and weak textures in large-view scenes, the geometric features of the scene are utilized in [25] to complete the regional correspondence of the scene and the multiple scales local correspondence of the same ratio. This type of traditional descriptor matching usually uses region priority matching or efficient sparse feature association, which is typically a direct matching scheme. But the robustness of this type of method decreased dramatically due to visual distortion occurring in large-view scenes; the localization performance is substantially reduced.

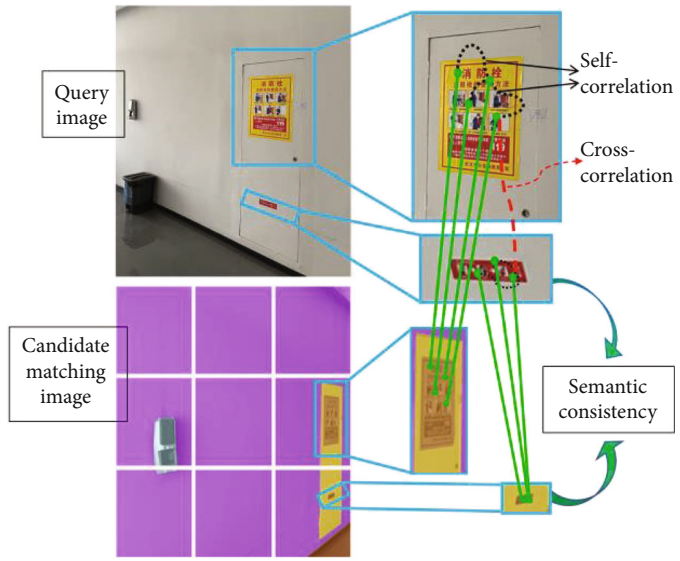


Figure 1: Schematic of sparse feature matching in weak texture scene with viewpoint change.

Camposeco et al. [26] proposed geometric outlier filtering to remove the mismatching relationship of features caused by a viewpoint change in large view scenes. The optimal camera pose estimation result in a large scene is the one with the most votes [27, 28], searching for the covisibility information between a query image and database images [7, 29, 30], retrieving the structural overlap region to eliminate the influence of the wide baseline scenes, and seeking the key frames and local matching features of query images [31], which can effectively remove the influence of viewpoint changes in the visual localization. To ensure the credibility of visual localization results, Taira et al. [10, 32] proposed the pose verification and incorporated scene geometric and semantic information for a trained pose verification model that generates a pose-score similar to the query image by a fractional regression convolutional neural network (CNN).

2.2. Visual Semantic Localization. Visual semantic features have richer scene information and object class information than traditional features, which is more robust to visual information distortion in large views [33]. In recent years, visual semantic information has been used in indoor positioning with promising results [34]. An extended structure-based method was proposed in [12] by combining image features and semantic understanding of the scene in the camera pose estimation stage of the query image. The method uses the geometric outlier filtering [27] and scene semantic labels to deal with the wide range of viewing scenarios where it is hard to seek the correct correspondences of image features. Toft et al. [35] proposed a sparse 3D point cloud model composed of scene curves and pixel-wise semantic labellings of the query image to enhance the image features discrimination for visual localization. Another semantic localization strategy is to include the image semantic information in the feature matching process of the visual localization algorithm [13, 36, 37], i.e., detecting and matching semantic features of the scene images. The latter type of semantic localization method only provides an additional weak semantic feature information does not solve the problem of seeking enough correct matches in wide baseline scenes, which motivates our work.

In contrast to the approaches previously discussed, our method focuses on the image feature matching stage of visual localization. Our model combines the sparse features in large-view scenes and the corresponding semantic information into a single confidence feature and learns discriminative and crosscorrelation of features, which completes accurate image matching to improve visual localization accuracy in wide baseline and long-range view scenes.

2.3. Learning-Based Feature Matching Network. Recent works show that the learning-based image matching network significantly improves matching performance [17]. Learning-based feature matching models can be divided into two categories. A common strategy of the first category is to learn the translation invariance and rotation invariance of feature descriptors [16, 38, 39] to enhance the representation of image features. A trainable single-image matching CNN was proposed in [40], which is a dense feature descriptor as well as a feature detector. The obtained keypoints by trainable CNN are more robust and stable than their traditional counterparts. The second category of approaches mainly focuses on different matching strategies for image features; for instance, a universal dense correspondence network was proposed [41] for geometric and visual semantic matching of images. Sarlin et al. [18] proposed a sparse feature matching model with GNN (graph neural network), which completes feature matching by self-attention and crossattention. A pixel dense matching network with a transformer was proposed in COTR [19], which selects query interest points and retrieves sparse counterparts between images to obtain local and global prior information by iteratively estimating scaling around the points. The same

self-attention and crossattention layers are used in LoFTR [17], a coarse-to-fine image matching model, where the steps of sequentially performing image feature detection, description, and matching are replaced by a pipeline using coarse-to-fine image feature matching. This algorithm conducts pixel dense matching at the coarse granularity and then refines the matching at fine granularity, which improves the image matching accuracy for weak texture scenes. However, if feature points are position-independent, they have similar background features (e.g., walls with weak texture or untextured corridors); some models [17–19] miscalculate the image attention weights and cannot complete accurate matching.

In contrast to the above, we focus on the precise correspondences of image features of the matching stage in visual localization. We propose multiscale forward attention to improve the self-correlation and crosscorrelation of sparse features for the anomalous scores of attention weighting of sparse feature points in large view scenes. We establish a coarse-to-fine feature matching model using a transformer network to better the image feature matching accuracy in extreme viewpoints.

3. Method

To address the matching ambiguity in the image matching phase of visual localization under large-view scenes, we propose a novel coarse to fine sparse feature matching network using a transformer, named MSFA-T, which is also suitable for other applications based on image matching such as object tracking and object retrieval. The structure of our model is shown in Figure 2, ε is the D2-Net [40] model used to extract the CNN features and the positions, $I - I'$ are the input images, and M is the mapping of feature map F and semantic segmentation map S.

Our goal is to train a coarse-to-fine sparse feature matching model that can output optimal geometric constraints for the visual localization algorithm in large-view scenes. First, we obtain the semantic features of the query image and candidate images by SETR [15], which can perceive the large view scene with failed localization in scene recognition. The scene semantic features are embedded into the sparse feature points of the image for coarse feature matching. Our model obtains the spatial locations of the feature points with similar semantics to learn the interrelationship of different feature points and decompose the similarity features, which completes a coarse matching of image features and the division of image patches with semantic classes and solves the problem of the misclassification of feature points under distorted view. We propose a multiscale forward attention mechanism (MSFA) embedded into a transformer to compute the attention weights of sparse features at different positions and to motivate the model to learn the self-correlation of features with the same semantic information and the crosscorrelation of features with different semantic information. MSFA module deals with the problem that the image distortion at a long-viewing distance produces anomalous scores on the attention weights of image features. The main specific constraints derived from the computation of feature vectors by the neural network (NN) are also executed. Finally, the transformer output vector is decoded by a multilayer perceptron (MLP) to obtain a confidence feature matrix for accurate image matching. Our model provides the optimal geometric constraints for visual localization.

After expanding the feature patches obtained by coarse matching to one-dimensional vector, we add positional encoding. We use the general linear positional encoding in transformers following DETR [42]; the positional encoding gives each feature patch unique position information to ensure that the transformed vectors of the sparse features become position dependent. This process enables our model to resist the influence of weak texture regions. The fused position-encoded feature vectors are fed into the transformer, and their weights are obtained according to our proposed multiscale forward attention module for computing confidence features.

3.1. Semantic Mapping for Coarse Matching. The mapping of semantic maps to image patches assigns different semantic labels to the sparse features of images, which facilitates the calculation of the self-correlation of feature points with the same semantic information and the crosscorrelation between different semantic feature points (as shown in Figure 1); meanwhile, it provides a priori information for coarse image matching. The incorrect matching of image features in weak texture scenes is significantly reduced. The specific computational details are as follows.

Semantic class labels are constructed by performing pixel-level semantic segmentation on all images [33], semantic label S_c is assigned to its same semantic class of image patches. The feature map after semantic mapping is defined as:

$$M = \left\{ \left(F_{i,j}, S_{c,(i,j)} \right) \right\}_{i=1,j=1}^{N}, \tag{1}$$

where each feature point and its image coordinates are defined as $F_{i,j}$, its semantic label class and corresponding semantic label position are defined as $S_{c,(i,j)}$, and N is the total number of feature patches.

As each feature patch is given a semantic label, we compare the observed feature patches and the corresponding semantic labels between the query and the database for scoring semantic consistency to obtain a coarse region where the feature patches are located. The semantic matching score is defined as

$$Y_{F_{i,j} \leftrightarrow F'_{i,j}} = \left\{ S_c \in \mathbb{R}^2, \left\| S_c \ominus S'_C \right\|^{-1} \right\}. \tag{2}$$

For the retrieved coarse match region, we crop it out with a partial window of size $m \times n$, as shown in the blue-boxed area in Figure 1. Coarse matching outputs of the same semantic region can be used with the dual-softmax operator, which is also the optimal transport layer in SuperGlue [18], as the output results can all be matched differentiable. The local window of a coarse matching region is refined to a

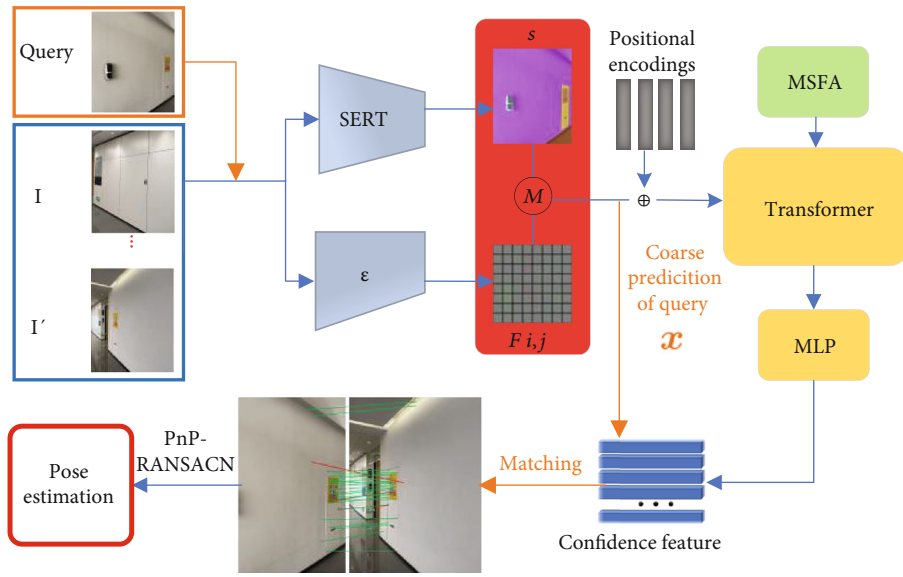


Figure 2: Overview of the proposed method, an image feature extraction, and matching structure.

subpixel level, the center of the query window is fixed as the query feature, and then, the distance between the feature in the window of the candidate image and the center of the feature in the query window is calculated; the image patches with highest scores are used as the accurate predicted match prediction of final image feature.

3.2. Multiscale Forward Attention Module. Feature matching models with a transformer calculate the attention weights of different feature points to enhance the correlation and uniqueness of feature points [17, 18], which reduces the feature matching errors in weak texture scenes. However, the type of methods are prone to abnormal scores in the process of local attention weights, which makes the larger deviation of relevant scores between neighboring feature points and leads to position-independent feature point matching errors in weak textures. We propose the multiscale forward attention module, MSFA, as shown in Figure 3. This module uses the self-attention weights at previous moment to smooth the anomaly scores at current moment and constrains the previous moment attention weights to optimize the forward attention model for the purpose of adaptive smoothing. A multiscale model, then, is introduced to different feature units for obtaining the target feature vectors with different characteristics, which solves the problem of attention weight anomaly scores.

We use different scale convolution filter (S-CF) on the basis of the multiheaded self-attention model to obtain feature units at different positions. We then model the feature units with different weights and calculate the interrelationships between different feature units. In addition, the target feature vectors with different weights are spliced and fused by a NN. Finally, the transformer output vector is decoded by a multilayer perceptron to obtain the confidence feature matrix. The specific calculation process is formulated as follows.

By smoothing the self-attention weights from the previous moment to the current moment, the new attention score of the current moment is $\bar{A}_{i,j}$.

$$\bar{A}_{i,j} = A_{i,j} \bullet \left(\sum_{t=0}^{l-1} \bar{A}_{i-t,j-1} \right), \tag{3}$$

where $A_{i,j}$ is the self-attention score at position i and moment j, $0 \le A_{i,j} \le 1$. The computation of attention weights is to select relevant information by measuring the similarity between query (Q) and each key (K); its output vector is a weighted sum of values with similarity scores. We use the dot product to weight the input features, which can be expressed as $A = \text{softmax}(Q \bullet K^T) \odot V$. l is the one-dimensional vector expanded by the input vector and positional encoding, and $\bar{A}_{i-t,j-1}$ is the forward attention score at the position $i - t$ of the previous moment.

After normalizing the forward attention weights at different positions using the softmax function, the anomalies at the current moment are smoothed by the self-attention weights to eliminate the anomalous scores of the attention weights, ensuring the continuity between the attention weights of different feature units at the previous and next moments. We note that the influence degree of single forward moment attention weight on n forward vectors is not consistent. It is not consistent for the attention weights of vectors at different moments; therefore, new constraint information needs to be added to the forward n vectors to improve the effect of smoothing anomalous attention score. We use a NN to generate a constraint factor φ_j to dynamically control the influence of the attention scores corresponding to different vectors in the previous moment on

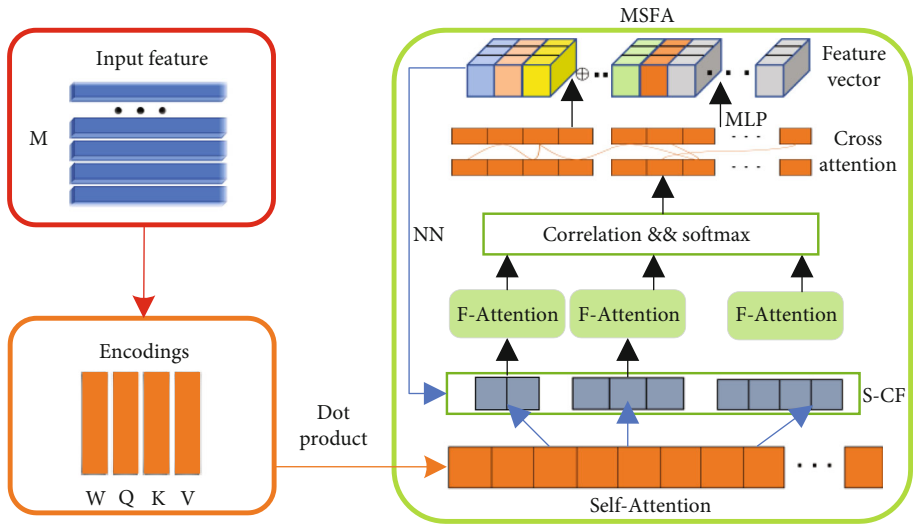


Figure 3: Multiscale forward attention module.

the different vectors in the current moment. The constraint factor φ_j is:

$$\varphi_j = NN\left(q_{j-1}, v_{j-1}, \bar{M}_{j-1}\right), \tag{4}$$

where q_{j-1} is the MLP decoder state at the previous moment, v_{j-1} is the target vector at the previous moment, $\bar{M}_{j-1}$ is the vectors sequence output from the decoder, and NN is a neural network model containing an implicit layer and a Sigmoid activation function. The constraint factor φ_j can add effective constraint information to the attention weight score at the previous moment; thus, the importance related to the vectors with higher attention anomaly scores will be reduced. By dynamically adjusting the importance of the attention weight score at the previous moment, one can optimally smooth that the abnormal attention score at the current moment is achieved. The smoothing function is shown by

$$\bar{A}_{i,j} = A_{i,j} \bullet \left(\sum_{t=0}^{l-1} \varphi_j \bullet \bar{A}_{i-t,j-1}\right). \tag{5}$$

The softmax function is used to normalize $\bar{A}_{i,j}$ so that the attention weights of vector units important at the previous moment are better learned at the current moment. Figure 4 shows the attention weights of the learned image features. Adaptive smoothing of the abnormal attention scores at the current moment is achieved by constraint factors to better align the vector positions of the model. MSFA-T assigns significant attention weights to the union distribution of sparse features in weak texture scenarios, which focuses on significant markings, structure information, object types, or feature location to learn the correlation of sparse feature points within the local regions of semantic consistency. It learns to ignore dynamic objects like pedestrians and repeated patterns like the corridor or wall.

The multiscale forward attention mechanism is used to solve the problem of anomalous attention weights of some feature vectors caused by a low degree of the model representation in weak texture scenes. Different from the multiheaded attention mechanism, we use different sizes of convolutional filters to calculate the respective scores of attention weights for each layer of the multiheaded attention model. The change patterns of feature vectors at different moments are obtained to model the vector units at different scales. Compared to using a single-scale filter in modeling the fixed vector units, the multiscale attention mechanism can extract deeper and richer feature information. In the multiscale model, convolution is computed for the forward attention score $\bar{A}_{j-1}$ using different sizes of convolution filters S-CF as follows.

$$f_j = \left\{C_k * \bar{A}_{j-1}, k = 1, \cdots, 4\right\}, \tag{6}$$

where C is the convolution operation and k is the convolution kernel size. As the image features are expanded as one-dimensional vectors and the positional encoding is also one-dimensional data, the one-dimensional convolutional filtering of different sizes corresponds to sliding windows of different sizes. Sliding on the vectors ensures that the vector units included each time can constitute a feature unit, thus preventing the same feature unit from being assigned different attention weights. The forward attention score of the convolution result f_j is calculated to obtain the target vector of K different feature units, which are finally stitched and integrated by one full connect to obtain the confidence feature matrix with more discriminative and correlative feature model representation.

With the MSFA module, not only can we get refined attention scores by modeling feature units at different scales but also smooth outliers by using normal attention scores from the previous moment to effective elimination of abnormal attention scores to complete the exact feature matching.

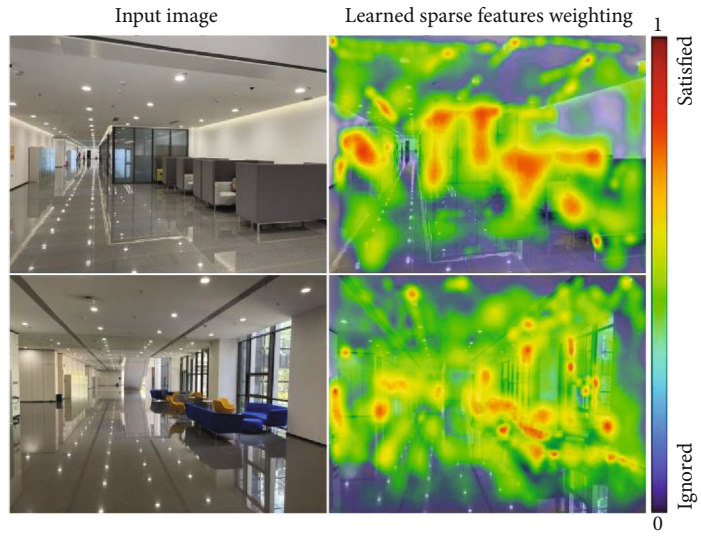


Figure 4: Heat maps of sparse feature attention weights in large view scenes.

3.3. Refined Matching and Loss Function. We obtain the exact matching prediction of a query image by selecting the matching terms based on the confidence threshold θ_m and mutual nearest neighbor (MNN) criteria. The matching process is as follows. We first calculate the score matrix Y between the output features by

$$Y_{i,j} = \tau^{-1}\left\{F_{i,j}, F'_{i,j}\right\}, \tag{7}$$

where the τ^{-1} is the scale factor of the matching, then softmax and MNN on the two dimensions of the score matrix Y are applied to predict the matching probability M_c. We denote the refined matching M_c as:

$$M_c = \{\forall(i,j) \mid MNN\left[\text{softmax}(Y_i, \cdot)_j \cdot \text{softmax}\left(Y_{\cdot,j}\right)_i\right] > \theta_m\}. \tag{8}$$

The final loss function includes both coarse-level loss and fine-level loss, i.e., $L = L_c + L_f$. In coarse matching, each feature point $F_{i,j}$ is directly compared for the score of semantic label consistency and distance difference; its generated variance $\sigma^2(i)$ is calculated by the position error to measure its uncertainty. The weighted loss function of the coarse-level matching is:

$$L_c = \sum_{i,j} \sigma^2(i)^{-1} \left\| S_c \ominus S'_C \right\|_2. \tag{9}$$

The fine-level loss function is generated from the negative log-likelihood loss on matrix M_c obtained by the dual-softmax operator. The feature matching is performed using MNN, so that the loss function is:

$$L_f = -\sum_{i,j} \log M_c(i,j). \tag{10}$$

In the localization phase, the output in feature matching with MSFA-T model is used to form a homography matrix using an efficient association algorithm for feature maps and 3D point clouds [30]. The pose of a query image is finally solved by PnP-RANSAC [9].

4. Implementation Details and Experiment Results

In this section, we present the training implementation details of our model, evaluate the image matching accuracy of MSFA-T compared with the state-of-the-art methods, and assess the role of the MSFA-T model in the visual localization systems.

4.1. Datasets. Training data: we train our image matching model MSFA-T on the ScanNet [43] dataset and the MegaDepth [44] dataset. ScanNet is an RGB-D indoor scene dataset that contains a series of views in 1513 indoor scenes annotated with 3D camera poses and semantic segmentations. MegaDepth dataset provides a large number of large-view images and corresponding dense depth maps generated by SFM [21], which includes large variations in appearance of scenes and viewpoint changes of a camera. The above datasets are required to learn translational invariance and rotational invariance models to improve the robustness of the model for large view scenes. Existing accuracy of the depth maps is sufficient to learn accurate local features [19] in the large view scenes, reducing the influence of weak texture scenes.

Testing data for image matching and visual localization: we used the image matching challenge HPatches dataset [45] to test the matching accuracy of our model for large-view scene images, as well as its robustness to viewpoint changes, long-view distance, and weak texture scenes. HPatches is a challenging dataset for image matching, which contains wide-baseline stereo images, long-range views images, and weak texture images. In addition, we used the InLoc [10]

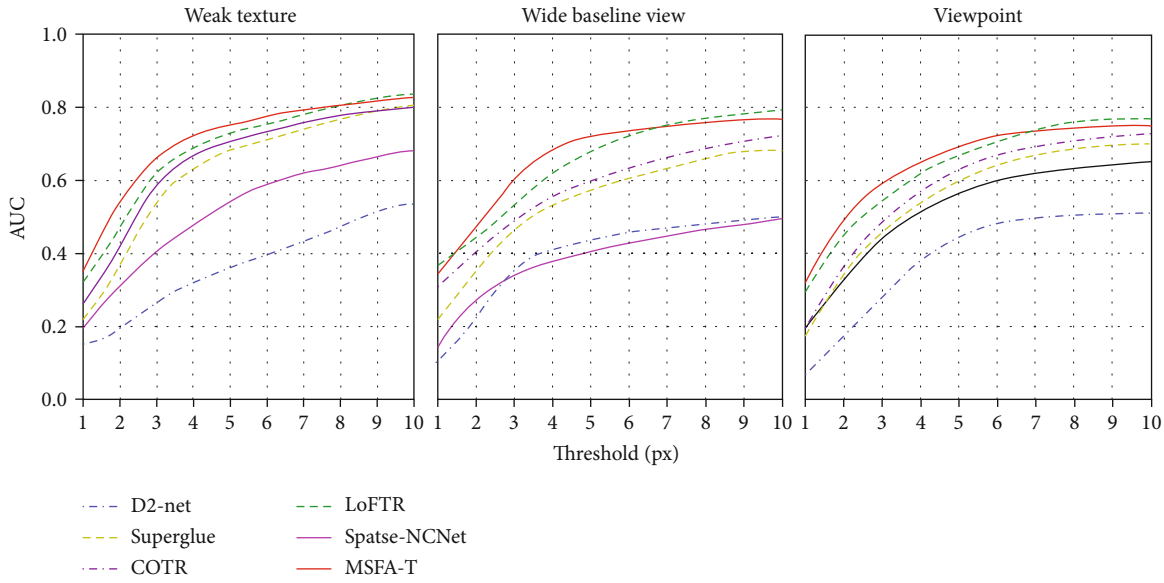


Figure 5: Evaluation on HPatches for image matching.

indoor dataset to test the improvement of MSFA-T in the accuracy of the localization algorithm and also used our previous work EfiLoc [9] and related localization algorithms, Image Bimodal Localization [46] and HAIL [47] for comparison. We compared only the image localization modules of the above localization algorithms. InLoc provides the large-scale indoor data based on two Washington University buildings, including 356 pieces of 4032×3024 query images and 9972 pieces of 1600×1200 database images that contain the scenes with wide baselines and weak textures. Thus, the localization based on such a dataset is a challenge considering the complexity of the indoor wide range of view scenes.

4.2. Implementation Details. We used a share backbone ResNet [48] architecture and a semantic segmentation SETR model [15] to initialize the CNN feature extraction network and semantic segmentation network, respectively. We used the feature map after the fourth downsampling layer of size $16 \times 16 \times 1024$ in the residual network with a convolutional kernel of size 1×1, the initial learning rate of $1 * 10^{-3}$ and a batch size of 64. For the transformer, we used the same number for layers of encoder and decoder; each encoder layer contains a self-attention layer and a multiscale forward-attention layer to ensure that accurate learning weights are assigned to each feature patch to enhance the self-correlation of image features. Each decoder layer contains the corresponding encoder-decoder attention layers without self-attention layers, which prevent the mutual communication between query points in order to enhance the relevant communication between query points and candidate points. Finally, we used 3-layer MLP to decode the vector output from the transformer and obtain the confidence matrix for query matching. We evaluate the performance of image-based visual localization systems [9, 46, 47] that use our image matching model and compared their localization accuracy under different scenarios.

Table 1: Evaluation on HPatches image pairs.

Method	AUC			#matches
	3 px	5 px	10 px	
D2-net	23.2	35.9	53.6	0.2 K
SuperGlue	53.9	68.3	81.7	0.6 K
COTR	62.8	67.9	80.6	1.0 K
Sparse-NCNet	48.9	54.2	67.1	1.0 K
LoFTR	65.9	75.6	**84.6**	1.0 K
MS-T	28.5	48.6	52.7	1.0 K
MFA-T	47.7	62.8	73.9	1.0 K
MSFA-T	**68.5**	**76.9**	83.5	1.0 K

4.3. Experiment Results. Image matching: to evaluate the performance of our model, we compared it with the state-of-the-art models, D2-Net [40], COTR [19], SuperGlue [18], Sparse-NCNet [49], and LoFTR [17]. D2-Net is a detector-based local feature matching network that uses a describe-and-detect methodology. The detection of D2-Net is postponed until a more reliable image feature is available and done jointly with the image description. SuperGlue is a detector-based local feature matcher, which uses self-attention and crossattention to improve the matching accuracy of image feature points (SuperPoint [16]). COTR, Sparse-NCNet, and LoFTR are detector-free matchers models, which have no local feature keypoints and directly output the dense matching result of the image. In addition, in order to confirm the important roles in assigning semantic features and multiscale forward attention mechanism to image CNN features in our model, we trained MS-T model, i.e., MSFA-T without multiscale forward attention mechanism, and MFA-T model, i.e., MSFA-T without semantic feature fusion module, respectively. We design ablation experiments to test their image matching performances in

Figure 6: Comparison results of image matching in large view scenes.

comparison with the related state-of-the-art matching algorithms.

For the matching challenge on the HPatches dataset, we restricted the number of image keypoints rather than the correct matching rate. For the image local feature matching algorithm, we restricted the extraction to a maximum of 2 K features with mutual nearest neighbors as the matches phase. For detector-free methods, which directly output the matches, we restricted the matches results with a maximum of 1 K outputting matches. Meanwhile, we used the initial default hyperparameters in the original matching algorithm implementation for all the baselines. Figure 5 shows the comparison of image matching results for wide-baseline view, weak texture, and viewpoint changes. For each method, we show the mean number of mutual nearest neighbor matches per image at different matching thresholds. From the comparison results, our method outperforms the other methods for the matching threshold below 7 pixels, especially in indoor scenes with weak textures and wide baseline views. Our approach makes the coarse-to-fine matching process that from semantic consistency matching to sparse features with the same semantic labels play an important role. Our multiscale forward attention overcomes the problem of anomalous scoring of sparse feature weights in weak texture scenes, which enhances the self-correlation and crosscorrelation of these features, improving the overall performance of the model.

The overall evaluation results on the HPatches dataset are shown in Table 1. We report the area under the cumulative curve (AUC) of corner error in image matching with the threshold of corner error being 3, 5, and 10 pixels, respectively. The AUC of the corner error as a function of the matching threshold in percentage is shown. Bold values in the table indicate the best results for that particular experiment. Our method has higher matching accuracy, especially for the error thresholds of 3 and 5 pixels in weak texture scenes.

Our MSFA-T matching model achieves the optimal performance with the error threshold values of 3 and 5 pixels, respectively. LoFTR achieves the optimal matching result with the error threshold value of 10 pixels because it uses the good matches at a fine level. In contrast, we fused the scene semantic features with the image CNN features so that the model filters out some semantic conflicting sparse features to ensure the refined matches of the images in complex large views.

We also perform ablation experiments on models MS-T and MFA-T. The MS-T model without the multiscale forward attention mechanism shows some sparse feature matching errors in weak texture scenes with wide baselines, which is due to the attention weight learning anomaly on position-independent features in this scene, causing the correlation between the features to be misallocated. The MFA-T model without the semantic feature fusion module shows the matching errors of some different types of objects due to the lack of the sparse features with semantic label information in wide baseline scenes and viewpoint change scenes. The MSFA-T model, which uses both the semantic information fusion mechanism and the multiscale forward attention mechanism, shows optimal matching results in large viewpoint scenes. The performances of the above models with the error threshold values of 3, 5, and 10 pixels, respectively, are shown in Table 1. These experimental results demonstrate the effectiveness of the coarse-to-fine network (semantic correspondence coarse matching to fine matching of features with the same semantic information) and multiscale forward attention mechanism proposed in this paper for refined image matching and also show the robustness of our method for large view scenes. The partial image matching schematic of our method with different module on indoor image pairs is shown in Figure 6. The green color indicates the correct match with a probability close to 1, in contrast, the lower the probability, the closer the color to red. MSFA-T achieves the best matches and fewer mismatches, which successfully copes with the image matching in weak texture areas and wide baseline views.

Indoor visual localization: accurate localization of indoor vision relies on robust image matching algorithms; therefore, we used the MSFA-T model in the image matching phase of indoor localization in indoor large view scenes to evaluate the localization performance of EfiLoc and related state-of-the-art visual localization algorithms [46, 47]. Similarly, we compare these localization algorithms using the MSFA-T model with original localization algorithms. EfiLoc-MSFA-T denotes the EfiLoc localization algorithm that uses the MSFA-T model, the same for others, e.g., IBL-MSFA-T

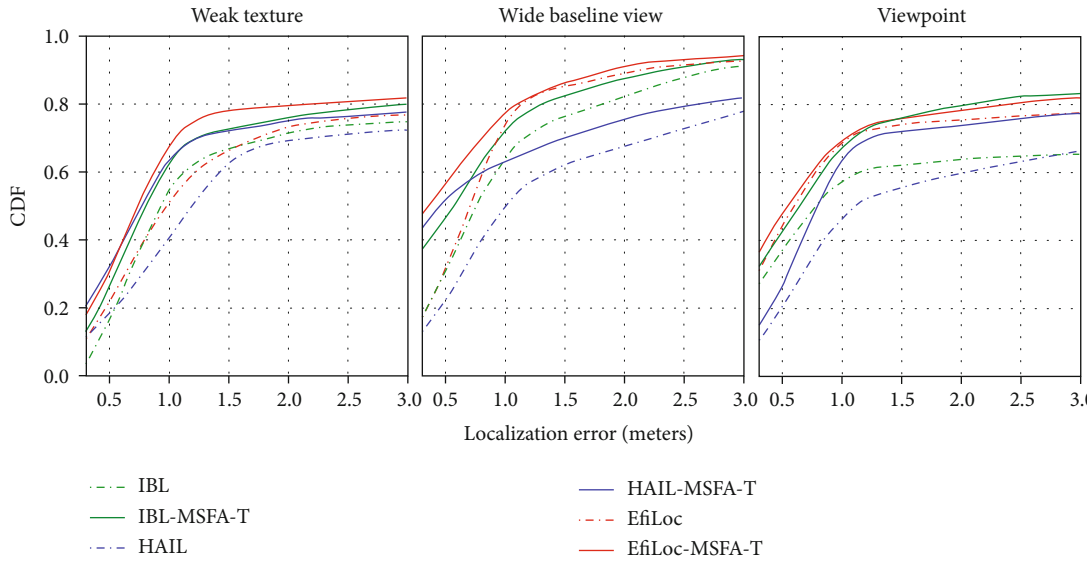


Figure 7: Localization error comparison.

denotes the Image Bimodal Localization with MSFA-T and HAIL with-MSFA-T (HAIL-MSFA-T). The comparison of the cumulative error function (CDF) of these positioning methods is shown in Figure 7.

The improvement of the visual localization performance is achieved by using the MSFA-T model instead of the image matching module in the original localization algorithm. The correct localized queries rate of the original localization algorithms (dashed lines in Figure 7) with the MSFA-T image matching model (solid lines) have different degrees of improvement with different influencing factors. At a localization error of 1 m, the performance improvement rate of the correct localized queries is 12% and 9% for IBL-MSFA-T and EfiLoc-MSFA-T, respectively. The general localization performance is most improved with HAIL. This is because HAIL uses the filtered SIFT feature keypoints that cannot accomplish robust image feature matching in the challenging scenarios described above, especially in indoor scenes with weak textures and viewpoint changes. This also demonstrates that our image matching model can successfully improve the performance of visual localization in large viewpoint scenes.

5. Conclusion

In this paper, we propose a model MSFA-T, a robust sparse feature matching network with a transformer, which accomplishes accurate image matching in visual localization in large view indoor scenes. MSFA-T successfully solves the problems of viewpoint distortion and weak textures using the image semantic information and the optimal confidence features. In addition, to deal with the problems of interrelationship and attention weight anomaly score of sparse feature points on different image patches, we use the transformer with our MSFA module for learning the specificity and correlation of the sparse features, which improves the matching accuracy of the sparse features in weak textures regions to enhance refined visual localization in large view scenes. MSFA-T accomplishes an average 79.8% probability of the AUC of the corner error in large view scenes, which outperforms the related state-of-the-art image matching algorithms. Moreover, our model improves on average the localization accuracy of image-based visual localization by 11.2% on the InLoc dataset. We believe the MFSA-T model takes a promising step toward refined image matching to improve a practical smartphone indoor localization services.

Data Availability

The data underlying the results presented in the study are available within the manuscript or directly access these publicly available datasets according to the references [10, 43–45].

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (grant no. 61971316).

References

[1] Y. Cao, R. Ji, L. Ji, G. Lei, H. Wang, and X. Shao, "l^2-MPTCP: a learning-driven latency-aware multipath transport scheme for industrial internet applications," *IEEE Transactions on Industrial Informatics*, 2022.

[2] J. Xu, E. Dong, Q. Ma, C. Wu, and Z. Yang, "Smartphone-based indoor visual navigation with leader-follower mode," *ACM Transactions on Sensor Networks (TOSN)*, vol. 17, no. 2, pp. 1–22, 2021.

[3] Z. Chen, A. Jacobson, N. Sünderhauf et al., "Deep learning features at scale for visual place recognition," in *2017 IEEE*

International Conference on Robotics and Automation (ICRA), pp. 3223–3230, May 2017.

[4] T. Sattler, A. Torii, J. Sivic et al., "Are large-scale 3d models really necessary for accurate visual localization?," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1637–1646, Honolulu,USA, July 2017.

[5] X. Bai, M. Huang, N. R. Prasad, and A. D. Mihovska, "A survey of image-based indoor localization using deep learning," in *2019 22nd International Symposium on Wireless Personal Multimedia Communications (WPMC)*, pp. 1–6, Lisbon,Portugal, November 2019.

[6] X. Chen and G. Fan, "Egocentric Indoor Localization From Coplanar Two-Line Room Layouts," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1549–1559, New Orleans, Louisiana, June 2022.

[7] Y. Li, N. Snavely, D. Huttenlocher, and P. Fua, "Worldwide pose estimation using 3d point clouds," in *European conference on computer vision*, pp. 15–29, Springer, Berlin, 2012.

[8] T. Sattler, B. Leibe, and L. Kobbelt, "Fast image-based localization using direct 2d-to-3d matching," in *International Conference on Computer Vision*, pp. 667–674, Barcelona Spain, November 2011.

[9] N. Li and H. Ai, "EfiLoc: large-scale visual indoor localization with efficient correlation between sparse features and 3D points," *The Visual Computer*, vol. 38, no. 6, pp. 2091–2106, 2022.

[10] H. Taira, M. Okutomi, T. Sattler et al., "InLoc: indoor visual localization with dense matching and view synthesis," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7199–7209, Salt Lake City, Utah, 2018.

[11] F. Gu, X. Hu, M. Ramezani et al., "Indoor localization improved by spatial context—a survey," *ACM Computing Surveys (CSUR)*, vol. 52, no. 3, pp. 1–35, 2020.

[12] C. Toft, E. Stenborg, L. Hammarstrand et al., "Semantic match consistency for long-term visual localization," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 383–399, Munich, Germany, October 2018.

[13] J. L. Schönberger, M. Pollefeys, A. Geiger, and T. Sattler, "Semantic visual localization," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6896–6906, Salt Lake City, Utah, 2018.

[14] N. Atanasov, S. L. Bowman, K. Daniilidis, and G. J. Pappas, "A unifying view of geometry, semantics, and data association in SLAM," *IJCAI*, pp. 5204–5208, 2018.

[15] S. Zheng, J. Lu, H. Zhao et al., "Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 6881–6890, 2021.

[16] D. DeTone, T. Malisiewicz, and A. Rabinovich, "Superpoint: self-supervised interest point detection and description," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pp. 224–236, Salt Lake City, UT, USA, December 2018.

[17] J. Sun, Z. Shen, Y. Wang, H. Bao, and X. Zhou, "LoFTR: detector-free local feature matching with transformers," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 8922–8931, 2021.

[18] P. E. Sarlin, D. DeTone, T. Malisiewicz, and A. Rabinovich, "Superglue: learning feature matching with graph neural networks," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4938–4947, 2020.

[19] Y. Xie, J. Zhang, C. Shen, and Y. Xia, "Cotr: efficiently bridging cnn and transformer for 3d medical image segmentation," *International conference on medical image computing and computer-assisted intervention*, pp. 171–180, Springer, Cham, 2021.

[20] X. Xin, J. Jiang, and Y. Zou, "A review of visual-based localization," in *Proceedings of the 2019 International Conference on Robotics, Intelligent Control and Artificial Intelligence*, pp. 94–105, Shanghai, China, September 2019.

[21] J. L. Schonberger and J. M. Frahm, "Structure-from-motion revisited," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4104–4113, Las Vegas, 2016.

[22] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[23] A. Alahi, R. Ortiz, and P. Vandergheynst, "Freak: fast retina keypoint," in *2012 IEEE conference on computer vision and pattern recognition*, pp. 510–517, Providence USA, June 2012.

[24] F. Youyang, W. Qing, Y. Yuan, and Y. Chao, "Robust improvement solution to perspective-n-point problem," *International Journal of Advanced Robotic Systems*, vol. 16, no. 6, article 172988141988570, 2019.

[25] X. Zuo, X. Xie, Y. Liu, and G. Huang, "Robust visual SLAM with point and line features," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1775–1782, Vancouver Canada, September 2017.

[26] F. Camposeco, T. Sattler, A. Cohen, A. Geiger, and M. Pollefeys, "Toroidal constraints for two-point localization under high outlier ratios," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4545–4553, USA, July 2017.

[27] L. Svärm, O. Enqvist, F. Kahl, and M. Oskarsson, "City-scale localization for cameras with known vertical direction," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 7, pp. 1455–1461, 2017.

[28] B. Zeisl, T. Sattler, and M. Pollefeys, "Camera pose voting for large-scale image-based localization," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2704–2712, Santiago, Chile, 2015.

[29] G. Lu and J. Song, "3D image-based indoor localization joint with WiFi positioning," in *Proceedings of the ACM on International Conference on Multimedia Retrieval*, pp. 465–472, New York, June 2018.

[30] T. Sattler, B. Leibe, and L. Kobbelt, "Efficient & effective prioritized matching for large-scale image-based localization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 9, pp. 1744–1756, 2017.

[31] P. E. Sarlin, C. Cadena, R. Siegwart, and M. Dymczyk, "From coarse to fine: robust hierarchical localization at large scale," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12716–12725, Long Beach, CA,USA, 2019.

[32] H. Taira, I. Rocco, J. Sedlar et al., "Is this the right place? Geometric-semantic pose verification for indoor visual localization," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4373–4383, Long Beach, USA, 2019.

[33] E. Stenborg, C. Toft, and L. Hammarstrand, "Long-term visual localization using semantically segmented images," in *In*

international conference on robotics and automation (ICRA), pp. 6484–6490, Brisbane Australia, May 2018.

[34] M. Sualeh and G. W. Kim, "Simultaneous localization and mapping in the epoch of semantics: a survey," *International Journal of Control, Automation and Systems*, vol. 17, no. 3, pp. 729–742, 2019.

[35] C. Toft, C. Olsson, and F. Kahl, "Long-term 3d localization and pose from semantic labellings," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pp. 650–659, Honolulu, HI, USA, 2017.

[36] N. Atanasov, M. Zhu, K. Daniilidis, and G. J. Pappas, "Localization from semantic observations via the matrix permanent," *The International Journal of Robotics Research*, vol. 35, no. 1-3, pp. 73–99, 2016.

[37] F. Yu, J. Xiao, and T. Funkhouser, "Semantic alignment of LiDAR data at city scale," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1722–1731, Boston, USA, 2015.

[38] P. H. Chen, Z. X. Luo, Z. K. Huang, C. Yang, and K. W. Chen, "IF-Net: an illumination-invariant feature network," in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 8630–8636, Paris, May 2020.

[39] K. M. Yi, E. Trulls, V. Lepetit, and P. Fua, "Lift: learned invariant feature transform," in *European conference on computer vision*, pp. 467–483, Springer, Cham, 2016.

[40] M. Dusmanu, I. Rocco, T. Pajdla et al., "D2-net: a trainable cnn for joint description and detection of local features," in *In Proceedings of the IEEE/cvf conference on computer vision and pattern recognition*, pp. 8092–8101, Long Beach, CA, USA, 2019.

[41] C. B. Choy, J. Gwak, S. Savarese, and M. Chandraker, "Universal correspondence network," *Advances in Neural Information Processing Systems*, vol. 29, 2016.

[42] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *European conference on computer vision*, pp. 213–229, Springer, Cham, 2020.

[43] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, "Scannet: richly-annotated 3d reconstructions of indoor scenes," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5828–5839, Honolulu, HI, USA, 2017.

[44] Z. Li and N. Snavely, "Megadepth: learning single-view depth prediction from internet photos," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2041–2050, Salt Lake City, UT, USA, 2018.

[45] V. Balntas, K. Lenc, A. Vedaldi, and K. Mikolajczyk, "HPatches: a benchmark and evaluation of handcrafted and learned local descriptors," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5173–5182, Honolulu, HI, USA, 2017.

[46] M. D. Redžić, C. Laoudias, and I. Kyriakides, "Image and wlan bimodal integration for indoor user localization," *IEEE Transactions on Mobile Computing*, vol. 19, no. 5, pp. 1109–1122, 2019.

[47] Q. Niu, M. Li, S. He, C. Gao, S. H. Gary Chan, and X. Luo, "Resource-efficient and automated image-based indoor localization," *ACM Transactions on Sensor Networks (TOSN)*, vol. 15, no. 2, pp. 1–31, 2019.

[48] 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, USA, 2016.

[49] I. Rocco, R. Arandjelović, and J. Sivic, "Efficient neighbourhood consensus networks via submanifold sparse convolutions," in *European conference on computer vision*, pp. 605–621, Springer, Cham, 2020.

Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 2117883, 9 pages
https://doi.org/10.1155/2022/2117883

Research Article

LiDAR: A Light-Weight Deep Learning-Based Malware Classifier for Edge Devices

Jinsung Kim,[1] **Younghoon Ban**,[1] **Geochang Jeon**,[2] **Young Geun Kim**,[3] **and Haehyun Cho**[2]

[1]*School of Software Convergence, Soongsil University, Seoul 06978, Republic of Korea*
[2]*School of Software, Soongsil University, Seoul 06978, Republic of Korea*
[3]*Department of Computer Science and Engineering, Korea University, Seoul 02841, Republic of Korea*

Correspondence should be addressed to Haehyun Cho; haehyun@ssu.ac.kr

Received 17 March 2022; Revised 15 May 2022; Accepted 2 June 2022; Published 14 June 2022

Academic Editor: Xun Shao

With the advent of the 5G network, edge devices and mobile and multimedia applications are used a lot; malware appeared to target edge devices. In the fourth quarter of 2020, 43 million pieces of malware targeting mobile devices occurred. Therefore, a lot of researchers studied various methods to quickly protect users from malware. In particular, they studied detecting malware for achieving the high accuracy with deep learning-based classification models on mobile devices. However, such deep learning-based classifiers consume a lot of resources, and mobile devices have limited hardware resources such as RAM and battery. Therefore, such approaches are difficult to be used in the mobile devices in practice. In this work, we study how a deep learning classifier classifies malware and proposed a novel approach to generate a light-weight classifier that can efficiently and effectively detect malware based on the insight that malware exhibits distinctive features as they are programmed to perform malicious actions such as information leaks. Therefore, by analyzing and extracting distinctive features used by a deep learning classifier from malicious dataset, we generate a light-weight rule-based classifier with high accuracy to efficiently detect malware on edge devices called LiDAR. On an edge device, LiDAR detects malware with 94% accuracy (F1-score) and 85.67% and 328.24% lower usages for CPU and RAM, respectively, than a CNN classifier, and showed the classification time 454.37% faster than the classifier.

1. Introduction

With the introduction of the 5G network, people enter the era of Internet of Things (IoT) in which more devices are connected as developed IoT; edge devices are growing a lot. It is expected that there will be more than 7.49 billion edge device (e.g., smartphones and wearable devices) users worldwide in 2025 [1]. Also, due to the high use of edge devices, multimedia applications are used a lot, and it is seen that cumulative downloads of multimedia applications (e.g., WhatsApp, YouTube, and Facebook) are about 28.4 billion or more [2]. Furthermore, mobile multimedia usage is about 4.23 hours per day which is consumed a lot of time [3]. Unfortunately, due to the severe security threats (e.g., Botnets and man-in-the-middle attack) and major privacy violations (e.g., social security numbers, credit card numbers, and passwords), the use of the edge devices is still risky [4–8]. For example, a single wrong click can launch a malicious program causing damage such as personal information leakage or financial loss. In the fourth quarter of 2020, 43 million pieces of malware targeting mobile devices appeared [9].

Such threats have led to the release of many commercial antivirus products such as Avast, Kaspersky, McAfee, and Norton. However, those antivirus products have a fatal limitation: They cannot detect unknown malware because they mostly rely on the signatures of known malicious applications [10]. To overcome the limitation, a lot

of research works have focused on developing malware detection approaches using deep learning algorithms to protect users [11–23].

Recently, along with the advances in mobile systems-on-a-chip (SoCs), there have been increasing pushes to run malware detection schemes directly on edge devices [11, 12]. This is because executing the schemes on the edge devices can improve the service response time by eliminating the data transfer overhead. It can save up to 46% overhead system consumption than local execution [24]. However, running deep learning-based malware detection approaches on edge devices is still at the nascent stage, since the edge devices are usually energy and resource constrained [25]. Running complex neural networks including many layers, nodes, and many features makes the edge devices consume CPU usage of at least 60% or more (six cores) and RAM usage of about 10 GB [26, 27]. Although previously studied deep learning-based malware detection approaches could achieve very high accuracy, it is hard to apply them on the edge device of which executing resources are limited. Consequently, it is of importance to develop approaches that can employ deep learning-based malware detection on the edge device.

In this work, we propose a novel approach to generate a deep learning-based light-weight classifier, named LiDAR, to enable efficient malware detection at the edge. To build the LiDAR, we first analyze malicious dataset such as SMS spam dataset, e-mail spam dataset, and Android malware dataset. We then extract word tokens from the malware dataset and train a convolutional neural network (CNN) algorithm using the extracted word tokens. Based on the trained CNN algorithm, we extract features that have high weight values using a visual explanation method of decisions from a large class of a CNN-based model, called gradient-weighted class activation map (Grad-CAM) [28], assuming that those features highly contribute to the prediction accuracy. Based on those features, we build a light-weight rule-based classifier.

To show the efficiency and effectiveness of LiDAR, we evaluate it on a workstation as well as the Raspberry Pi. Our evaluation results clearly demonstrate that LiDAR significantly improves the resource utilization as well as the classification time, compared to the state-of-the-art CNN-based classifiers, achieving the feasible accuracy: on average, LiDAR showed 85.67% and 328.24% lower usages for CPU and RAM, respectively, than a CNN classifier, and showed the classification time 454.37% faster than a CNN classifier to detect Android malware, while achieving 93% of the prediction accuracy.

In summary, our contributions are as follows:

(i) First, we analyze general approaches of malware detection process using deep learning-based classification models with spam dataset and Android application dataset

(ii) Second, based on the analysis, we use a deep learning algorithm to find distinctive features of malware. And, we design a light-weight classifier with the high accuracy to efficiently detect malware on edge devices

(iii) Lastly, we thoroughly evaluate a prototype of LiDAR. Also, we compare our classifier against deep learning classifiers to demonstrate the computation resources and classification time of it. Our approach shows better performance of 85.67% and 328.24% lower usages for CPU and RAM than CNN classifiers with 94% accuracy (F1-score)

2. Background and Related Work

In this section, we introduce the advantages and disadvantages of Android malware detection using deep learning-based approaches. We, also, discuss commonly used features of Android malware employed by the previous studies.

2.1. A Limitation of Deep Learning-Based Android Malware Classification Approaches. Recently, a surge of studies were proposed to detect Android malware by using deep learning-based approaches using various features [11–23]. The advance of deep learning algorithms helps achieve the high accuracy by learning distinctive features of data with complex neural networks. Table 1 shows the accuracies (or F1-score) of previous deep learning-based malware detection approaches with algorithms and features used. However, classifiers generated by deep learning algorithms usually require the high computation time and resource usage because many approaches use excessive and detail features based on complex neural networks to achieve the high accuracy [11–19]. Consequently, even though they could achieve the high accuracy, it is difficult to employ them in practical on the most of smart edge devices which have limited computing resources.

2.2. Commonly Used Features for Android Malware Detection. Table 1 summarizes state-of-the-art deep learning-based malware detection approaches. In general, the methods are built based on various features including permissions and/or API calls. Permissions include information on the system-level functionalities, such as current location and network status. API calls are related to the functionalities that an application provides to users (e.g., SMS functions, call functions, and read and write functions). Malicious applications usually exploit specific permissions or API calls, such as reading sensitive data (e.g., a function reading a password) or transferring data (e.g., a function writing to a socket), to leak private data or capture the user behaviors. By using combinations of such features, previous approaches aimed to not only detect malware but also discover its malicious behaviors to assist the wholistic analysis process.However, in edge use cases, it does not necessarily use such detailed features because we merely need to discover whether an application is malicious or not rather than discovering its malicious behaviors in detail. Also, malicious applications usually share distinct features because they are programmed to inflict damages such as sensitive information leaks or financial loss to users. Hence, based on this insight, we propose a way to

Table 1: Summary of deep learning-based malware classification approaches.

Name	Algorithm	Accuracy or F1-score	Features
MalDozer [11]	CNN	96%	API call
DL-Droid [12]	MLP	99%	Permission, etc.
Droid-Sec [13]	DBN	97%	Permission, API call, etc.
Kim et al. [14]	DNN	99%	Permission, component, string, opcode, API
DroidDetector [15]	DBN	97%	API, permission, etc.
DroidDeep [16]	DBN	99%	Permission, API call, action, component, etc.
Li et al. [17]	DNN	97%	Permission, API call, etc.
Ganesh et al. [18]	CNN	93%	Permission
Nix and Zhang [19]	CNN	99%	API call

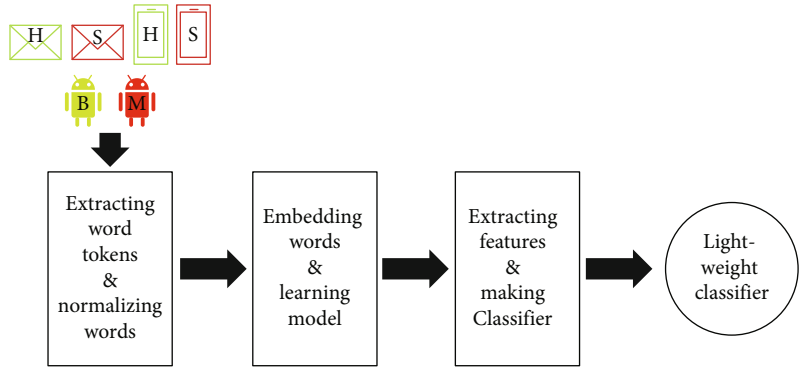


Figure 1: The overview of our approach to generate a deep learning-based light-weight classifier.

Table 2: The summary of our dataset.

	Malicious data				Benign data			
	Spam SMS	Spam e-mail	Android malware		Spam SMS	Spam e-mail	Android malware	
			2019	2020			2019	2020
Training dataset	600	2,953	1,600	1,600	3,857	5,575	1,600	1,600
Test dataset	147	768	400	400	968	1,364	400	400

generate a light-weight classifier that can efficiently detect malicious applications.

3. Overview

We first analyze how deep learning-based classifiers classify malware. Based on the analysis, we aim to design an approach to generate a light-weight classifier with the high accuracy to efficiently detect malware on edge devices. To achieve the goal, we employ a deep learning algorithm to find distinctive features of malware. Since we cannot directly obtain the distinctive features from the trained neural network due to its insufficient explainability, we use Grad-CAM that visualizes how much the features contribute to the classification accuracy. Based on the extracted distinctive features, we build a light-weight rule-based classifier, named LiDAR. It is worth noting that our approach can be applied onto the malware classification problem as well as other types of data which have remarkable features such as scam email. In general, such "malicious" samples in any dataset have distinguishable features from benign samples because attackers create them to have uncommon features shared by benign samples. Therefore, by using distinctive features from malware, we could reduce features and lowering overhead classification for malware detection.

In the following sections, we show how we collected the dataset for this study (in Section 4.1), how we preprocess the dataset (in Section 4.2), how we learn features of the dataset by using a deep learning algorithm (in Section 4.3), how we select important features with a visual explanation technique from the deep learning-based model (in Section 4.4), and how we generate a light-weight classifier based on the features (in Section 4.5).

4. Design

In this section, we demonstrate our approach to generate a light-weight classifier based on the learning result of a deep

Table 3: Malicious and benign features discovered by Grad-CAM.

(a)

SMS spam dataset			E-mail spam dataset		
No.	Weight value	Features	No.	Weight value	Features
1	0.0023	call	1	0.0159	click
2	0.0016	free	2	0.0141	run
3	0.0016	www	3	0.0088	could
4	0.0014	stop	4	0.0086	file
5	0.0013	txt	5	0.0074	remov
6	0.0013	repli	6	0.0070	modem
7	0.0010	cash	7	0.0068	send
	…			…	
800	-0.0017	see	31,860	-0.0144	link
801	-0.0017	heart	31,861	-0.0178	make
802	-0.0018	give	31,862	-0.0237	one
803	-0.0018	weekend	31,863	-0.0334	nbsp
804	-0.0031	get	31,864	-0.0454	emailaddr
805	-0.0038	got	31,865	-0.1133	httpaddr

(b)

Android malware dataset					
2019				2020	
No.	Weight value	Features	No.	Weight value	Features
1	0.0118	android.app-> android.view	1	0.0180	android.app-> android.view
2	0.0094	android.content-> android.content	2	0.0142	android.view-> android.content
3	0.0088	android.content-> android.app	3	0.0135	android.os-> java.lang
4	0.0082	android.webkit-> java.lang	4	0.0130	android.content-> java.lang
5	0.0081	android.app-> android.content	5	0.0124	android.content-> android.app
6	0.0068	android.app->android.os	6	0.0094	android.view-> android.view
7	0.0058	android.view-> android.view	7	0.0092	android.net-> android.net
…				…	
4,667	-0.0051	java.net->java.lang	13,201	-0.0018	android.content.res-> java.lang
4,668	-0.0056	android.view->java.lang	13,202	-0.0023	android.database.sqlite-> android.database.sqlite
4,669	-0.0064	java.io->java.io	13,203	-0.0025	android.webkit-> android.util
4,670	-0.0065	android.widget-> java.lang	13,204	-0.0064	java.io->java.io
4,671	-0.0069	android.content-> android.os	13,205	-0.0065	android.view-> android.util
4,672	-0.0081	android.widget-> android.util	13,206	-0.0068	android.widget-> android.util

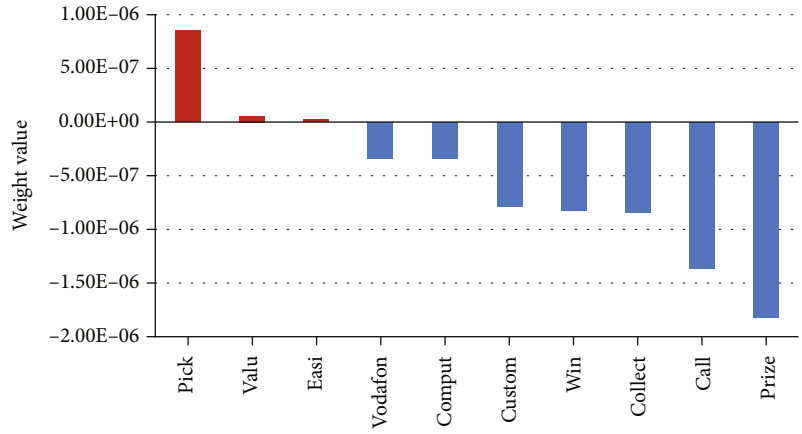


Figure 2: Examples of weights obtained from the SMS spam dataset by using Grad-CAM.

Table 4: The number of word tokens used in our experiments. M: malicious features; B: benign features.

	SMS spam [37]		E-mail spam [38]		Android malware [4] 2019		2020	
CNNc	6,272		82,005		9,211		18,925	
CNNg and LiDAR	M	B	M	B	M	B	M	B
	428	337	12,382	19,483	2,644	2,028	6,576	6,630

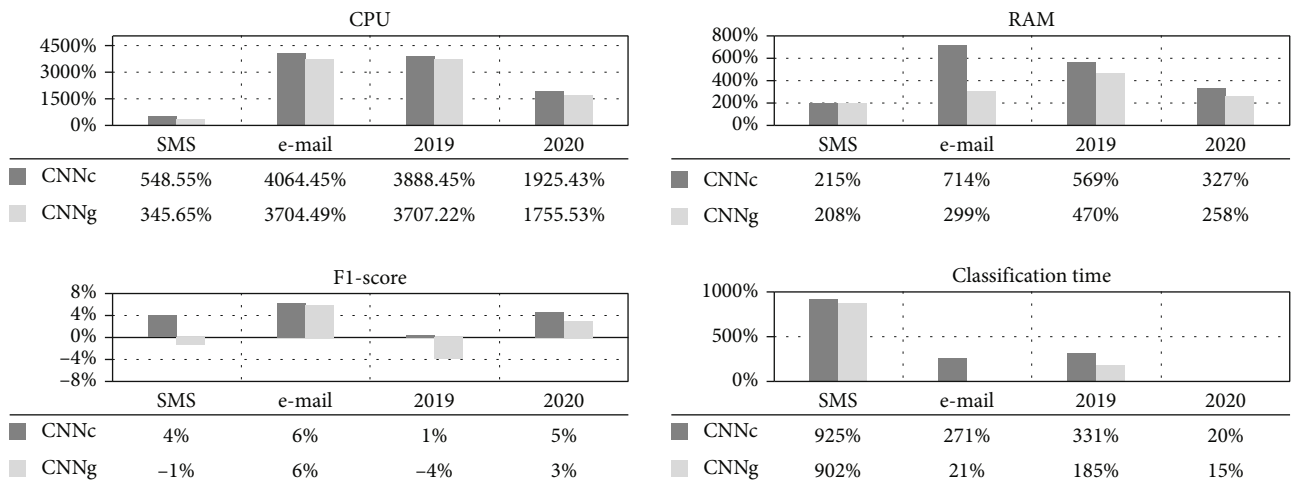


Figure 3: The comparison of F1-scores and the performance overhead of the CNN-based classifiers on the workstation based on the evaluation results of LiDAR.

learning algorithm to classify data samples that have distinctive features such as malware. Figure 1 shows the overview of our approach.

4.1. Dataset. In this work, we collected 24,232 real-world data as in Table 2, which consists of SMS spam message dataset [29], e-mail spam dataset [30], and Android malware dataset appeared from 2019 to 2020 [4]. By using our dataset, we demonstrate that malicious samples of the dataset have notable features to distinguish them from benign data samples, and thus, we can generate a much lighter classifier than deep learning-based models.

4.2. Preprocessing. To make light-weight classifiers, we use word tokens. We, thus, transform the malicious dataset (i.e., SMS spam dataset, e-mail spam dataset, and Android malware dataset) to word tokens. Finally, we remove duplicated word tokens.

4.2.1. Word Normalization. To remove unnecessary texts such as special characters, newline, and stopword for malware classification, we normalize the dataset. We, then, group texts that means the same (e.g., abc@abc.com to email address, https (http) to http address, and $ to dollar). On the other hand, Android malware datasets have many text

Table 5: The evaluation results on the workstation using the three classifiers.

Dataset	Classifier	CPU (%)	RAM (MB)	Classification time (seconds)	F1-score
SMS	CNNc	245.80%	262.13	0.96	0.94
	CNNg	168.90%	256.27	0.94	0.89
	LiDAR	37.90%	83.30	0.09	0.90
E-mail	CNNc	4,451.80%	1,986.491	14.04	0.99
	CNNg	4,067.00%	972.08	4.57	0.98
	LiDAR	106.90%	243.92	3.78	0.93
Malware in 2019	CNNc	3,868.80%	980.04	1.85	0.94
	CNNg	3,693.00%	835.86	1.22	0.90
	LiDAR	97.00%	146.55	0.43	0.94
Malware in 2020	CNNc	3,862.50%	944.33	1.57	0.99
	CNNg	3,538.50%	792.52	1.51	0.97
	LiDAR	190.70%	221.27	1.31	0.94

Table 6: The evaluation results on the Raspberry Pi using the three classifiers.

Dataset	Classifier	CPU (%)	RAM (MB)	Classification time (seconds)	F1-score
SMS	CNNc	176.00%	264.21	3.31	0.94
	CNNg	169.00%	256.54	3.28	0.89
	LiDAR	84.70%	111.01	0.34	0.90
E-mail	CNNc	353.00%	2,029.13	130.73	0.99
	CNNg	345.80%	870.11	45.35	0.98
	LiDAR	167.50%	280.37	18.44	0.93
2019	CNNc	306.70%	582.38	7.51	0.94
	CNNg	294.10%	515.59	5.94	0.90
	LiDAR	189.60%	166.84	1.71	0.94
2020	CNNc	305.40%	638.52	8.15	0.99
	CNNg	303.40%	582.70	6.99	0.97
	LiDAR	172.80%	262.40	6.51	0.94

features (in Section 2.2). Hence, we use Android framework APIs as the main feature of Android malware. We, also, extract API call graphs (ACG) by which we can track data flows between a point where sensitive data is read and another point where the sensitive data is exported by using FlowDroid [31].

4.2.2. Word Encoding for the Malware Dataset. To learn the malware dataset, we convert a preprocessed each word token in the malware dataset to an integer number for the efficiency. When we meet unknown tokens that could not find in the learning process, we map such word tokens to "Unknownword" token. Lastly, add paddings to make the malware dataset the same length.

4.3. CNN Architecture. We employ a simple CNN for the deep learning algorithm [32, 33]. CNN is widely used to find common features of malware word tokens that are frequently used in actual malware dataset [34]. We use a standard convolutional neural network architecture. The input first goes through an embedding layer and then a one-dimensional convolutional layer (Conv1D) with ReLu activations. The last layer is a dense layer after we flattened data into a vector. The Conv1d is trained by a word using kernel size of 1 to capture a feature of each. We also use the Sigmoid activation function, to further classify binary labels.

4.4. Feature Selection. To investigate how different word token features contribute to the accuracy of a CNN classifier, we use Grad-CAM. Grad-CAM enables one to visualize each feature map layer and understand how the input data of a CNN affect the classification. Also, Grad-CAM can extract weight values without architectural changes or retraining. Grad-CAM exploits the feature maps extracted from the Conv1D layers to identify the impact of the features on the classification results. Grad-CAM sorts the feature maps based on the weight values of any class flowing into the final convolutional layer. As a result, Grad-CAM can extract a heat map of weight values for the word tokens which can be used for the light-weight classification.

Table 3 shows extracted features of the malware dataset using Grad-CAM. Higher values indicate malicious features, while lower values indicate benign features.

4.5. LiDAR. To build the light-weight classifier, we identify important features to classify malware from the malware

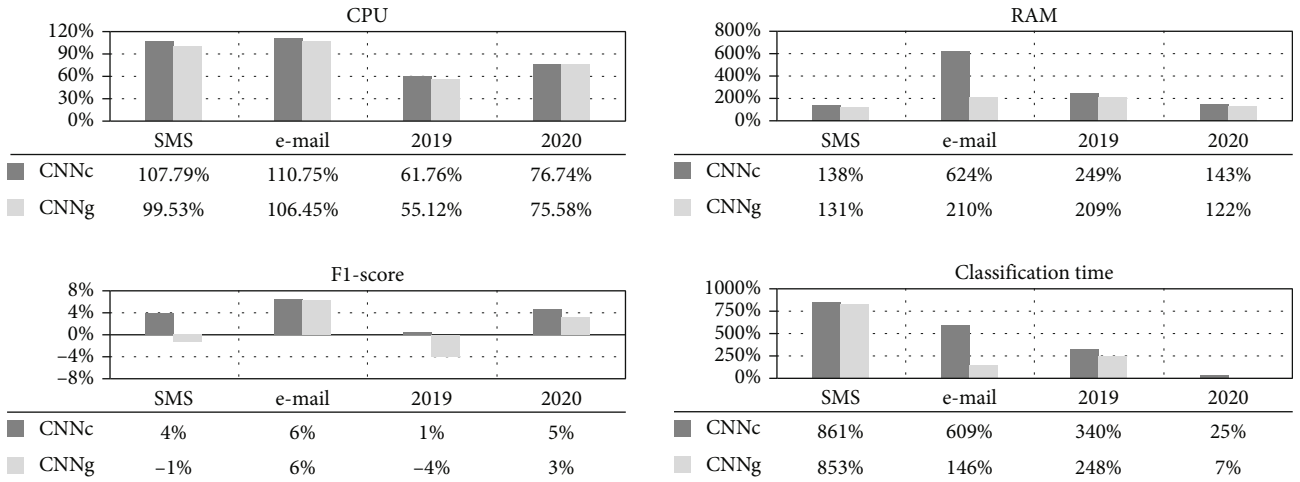


Figure 4: The comparison of F1-scores and the performance overhead of the CNN-based classifiers on the Raspberry Pi based on the evaluation results of LiDAR.

dataset based on the weight values of the extracted features (Section 4.4). As a running example, Figure 2 shows the classified malicious data from the SMS spam dataset based on the weighted values by using the CNN algorithm. In Figure 2, the first three words indicate malicious weighted values, and the others indicate benign weighted values. In this case, an average of more than one-third of the 600 training SMS spam dataset can be identified as the malicious weight values. This means that the malware dataset has more than one-third of distinct malicious features, and the malware dataset can be classified by the number of malicious values. The rule-based classifier can be built based on the observation, by analyzing the number of malicious weight values. Because the CNN classifier does not classify the malware with the context information of SMS spam dataset but with the observed number of distinct words, the rule-based classifier can be built using the following two conditions: (i) When a data has a lot of prelearned words—in this case, we can apply a heuristic condition when a data do not have more than 1/3 of prelearned malicious or benign words. If a data sample has more than one-third of malicious words, we classify it as malware. (ii) On the other hand, if a data sample contains more malicious words than benign words, we classify it as malware. By exploiting distinctive features of malware, we can generate an effective classifier much lighter than a deep learning classifier, albeit we need manual efforts to decide the threshold for classifying malware.

5. Evaluation

In this section, we evaluate our approach to demonstrate its efficiency and effectiveness. We use a Raspberry Pi using the ARM64 architecture as well as a workstation. For the convenience, we refer the CNN classifier to CNNc, CNN classifier using high-weight features to CNNg, and our approach to Light-weight Deep Learning-based Malware Classifier (LiDAR).

5.1. Experiment Setup. We performed our evaluations on a workstation running Ubuntu 18.04 with 20-core Intel Xeon Gold 6230 two CPUs at 2.10 GHz, 256 GB RAM, and a NVIDIA GeForce RTX 2080 GPU. And we conduct experiments on a Raspberry Pi 4 Model B (Rev 1.4) running Ubuntu 18.04 with a 4-core Cortex-A72 (ARM v8), 4GB RAM. We implemented LiDAR by using Python v3.7.1, TensorFlow GPU v1.14.0, Keras v2.2.4, CUDA v11.2, and FlowDroid v1.5 for extracting ACG.

Table 4 shows that the number of words used for performance comparison in each classifier.

5.2. Evaluation Metrics. To explore the effectiveness and efficiency, we used the following metrics.

(1) *CPU Usage*. We consider the maximum workload that a single CPU can handle is 100%, and we show the classifier's CPU usage based on it (e.g., if CPU usage is 200%, it means we need two cores fully to perform a classification)

(2) *RAM Usage*. We measure the resident set size (RSS) of a classifier when it runs

(3) *Classification Time*. We measure the total execution time of a classifier

(4) *F1-Score*. We use the F1-score of classification results to show the effectiveness of each classifier

5.3. Evaluation Results on the Workstation. In this section, we evaluate classifiers on a workstation using malware dataset (SMS spam dataset, e-mail spam dataset, Android malware dataset).

Figure 3 and Table 5 show the experimental results. CNNc used an average of 3,107% of the CPU usage, and CNNg used an average of 2,867%. On the other hand, LiDAR showed an average of 108% of the CPU usage, which is much lower than the CPU usage of CNNc and CNNg. In addition, the RAM usage of LiDAR is also averagely 500.4%

and 311.02% lower than that of CNNc and CNNg, respectively, as shown in Table 4. These results yielded the significant improvement of classification time of LiDAR (averagely 228% and 46.78% faster than CNNc and CNNg, respectively). Nevertheless, LiDAR achieves almost similar F1-score with CNNs and CNNg; the accuracy difference of CNNc and CNNg is only 3.87%. These results imply that LiDAR strikes a good trade-off point between the performance and prediction accuracy.

5.4. Evaluation Results on the Raspberry Pi. Table 6 and Figure 4 illustrate evaluation results of each classifier on the Raspberry Pi. CNNc and CNNg used 285% and 278% CPU usages on average, but the CPU usage of LiDAR is 154% on average, which is 80.98% and 85.67% lower than the CPU usage of CNNc and CNNg, while the RAM usage of CNNc and CNNg is 328.24% and 171.13% on average, which is much higher than that of LiDAR. As a result, LiDAR has an average classification time of 454.37% and 127.95% faster than CNNc and CNNg. Despite the improvement of these results, there is only a small difference in F1-score of 3.87% with CNNs and CNNg, such as the experimental results on a workstation. Consequently, we can observe that LiDAR offers a good compromise between the performance and classification accuracy in any environment.

6. Conclusion

With the advent of the 5G network, a lot of malware targeting IoT devices occurred. Accordingly, a lot of research is on deep learning-based approaches to quickly protect users from malware. However, such deep learning-based approaches consume a lot of resources. In this work, to enable efficient malware detection on the edge devices, we proposed a novel approach to generate a light-weight classifier, LiDAR. We analyzed the SPAM and malware features by using deep learning-based Grad-CAM. Based on distinct features extracted by Grad-CAM, we built LiDAR with a rule-based classifier. Our evaluation results show that LiDAR can effectively detect malware achieving 92.78% of prediction accuracy, while only exhibiting 154% and 205.15 MB of CPU and memory resources, respectively, which resulted in the significant improvement in the classification time: roughly two times faster than a CNN-based deep learning model on average.

6.1. Limitations and Future Works. First off, LiDAR has the out of vocabulary problem as the other deep learning-based approaches have. If our classifier meets an unknown word token, the token is simply ignored. Therefore, to use LiDAR in practice, it is important to continuously learn emerging malware. In addition, similar to the other malware classification approaches, LiDAR cannot detect heavily obfuscated malware because we cannot find effective word tokens from malware if obfuscation techniques such as the class encryption are applied on the malware. We note that classifying unknown and obfuscated malware is a challenging problem, and the limitation is common in deep learning-based approaches. We leave these limitations as future work.

Data Availability

The data used to support the findings of this study were supplied by Jinsung Kim under license and so cannot be made freely available. Requests for access to these data should be made to Jinsung Kim (okokabv@soongsil.ac.kr).

Conflicts of Interest

The authors declare that they have no conflict of interest.

Acknowledgments

This work was supported by the National Research Foundation of Korea (NRF) Grant through the Korean Government (MSIT) under Grant NRF-2021R1A4A1029650.

References

[1] L. S. Vailshery, "Number of connected wearable devices worldwide by region from 2015 to 2022," 2021, https://www.statista.com/statistics/490231/wearable-devices-worldwide-by-region.

[2] L. Ceci, "Most downloaded mobile apps worldwide from 1st quarter 2014 to 3rd quarter 2021," 2021, https://www.statista.com/statistics/1280313/downloads-top-apps-worldwide/.

[3] L. Ceci, "Time spent per day with mobile non-voice media in the United States from 2019 to 2023," 2022, https://www.statista.com/statistics/469983/time-spent-mobile-media-type-usa/.

[4] VirusShare, "Android malicious applications dataset," 2021, https://virusshare.com/.

[5] K. W. Ching and M. M. Singh, "Wearable technology devices security and privacy vulnerability analysis," *International Journal of Network Security & Its Applications*, vol. 8, no. 3, pp. 19–30, 2016.

[6] A. D. Raju, I. Y. Abualhaol, R. S. Giagone, Y. Zhou, and S. Huang, "A survey on cross-architectural IoT malware threat hunting," *IEEE Access*, vol. 9, pp. 91686–91709, 2021.

[7] M. Al-Hawawreh, F. den Hartog, and E. Sitnikova, "Targeted ransomware: a new cyber threat to edge system of brownfield industrial Internet of Things," *IEEE Internet of Things Journal*, vol. 6, no. 4, pp. 7137–7151, 2019.

[8] H. Haddadpajouh, A. Mohtadi, A. Dehghantanaha, H. Karimipour, X. Lin, and K.-K. R. Choo, "A multikernel and metaheuristic feature selection approach for IoT malware threat hunting in the edge layer," *IEEE Internet of Things Journal*, vol. 8, no. 6, pp. 4540–4547, 2020.

[9] McAfee, "Labs Mobile Threat Report," 2021, https://www.mcafee.com/content/dam/global/infographics/McAfeeMobileThreatReport2021.pdf.

[10] I. Santos, F. Brezo, X. Ugarte-Pedrero, and P. G. Bringas, "Opcode sequences as representation of executables for data-mining-based unknown malware detection," *Information Sciences*, vol. 231, pp. 64–82, 2013.

[11] E. B. Karbab, M. Debbabi, A. Derhab, and D. Mouheb, "MalDozer: automatic framework for android malware detection using deep learning," *Digital Investigation*, vol. 24, pp. S48–S59, 2018.

[12] M. K. Alzaylaee, S. Y. Yerima, and S. Sezer, "DL-Droid: deep learning based android malware detection using real devices," *Computers & Security*, vol. 89, p. 101663, 2020.

[13] Z. Yuan, Y. Lu, Z. Wang, and Y. Xue, "Droid-sec: deep learning in android malware detection," *ACM conference on SIGCOMM*, p. 2014, 2014.

[14] T. Kim, B. Kang, M. Rho, S. Sezer, and E. G. Im, "A multimodal deep learning method for android malware detection using various features," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 3, pp. 773–788, 2019.

[15] Z. Yuan, Y. Lu, and Y. Xue, "Droiddetector: android malware characterization and detection using deep learning," *Tsinghua Science and Technology*, vol. 21, no. 1, pp. 114–123, 2016.

[16] X. Su, D. Zhang, W. Li, and K. Zhao, "A deep learning approach to android malware feature learning and detection," *IEEE TrustCom-BigDataSE-ISPA*, p. 2016, 2016.

[17] D. Li, Z. Wang, and Y. Xue, "Fine-grained android malware detection based on deep learning," *IEEE Conference on Communications and Network Security (CNS)*, 2018, pp. 1-2, Beijing, China, 2018.

[18] M. Ganesh, P. Pednekar, P. Prabhuswamy, D. S. Nair, Y. Park, and H. Jeon, "CNN-based android malware detection," in *International Conference on Software Security and Assurance (ICSSA)*, pp. 60–65, Altoona, PA, USA, 2017.

[19] R. Nix and J. Zhang, "Classification of android apps and malware using deep neural networks," in *International joint conference on neural networks (IJCNN)*, pp. 1871–1878, Anchorage, AK, USA, 2017.

[20] V. Sihag, M. Vardhan, P. Singh, G. Choudhary, and S. Son, "PICAndro: packet inspection-based android malware detection," *Journal of Internet Services and Information Security (JISIS)*, vol. 2021, no. 2, pp. 1–11, 2021.

[21] J. Jung, H. Kim, S. Cho, S. Han, and K. Suh, "Efficient android malware detection using API rank and machine learning," *Journal of Internet Services and Information Security (JISIS)*, vol. 9, no. 1, pp. 48–59, 2019.

[22] A. L. Marra, F. Martinelli, F. Mercaldo, A. Saracino, and M. Sheikhalishahi, "D-BRIDEMAID: a distributed framework for collaborative and dynamic analysis of android malware," *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications (JoWUA)*, vol. 11, no. 3, pp. 1–28, 2020.

[23] R. Casolare, C. De Dominicis, G. Iadarola, F. Martinelli, F. Mercaldo, and A. Santone, "Dynamic mobile malware detection through system call-based image representation," *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications (JoWUA)*, vol. 12, no. 1, pp. 44–63, 2021.

[24] I. A. Elgendy, W.-Z. Zhang, Y. Zeng, H. He, Y.-C. Tian, and Y. Yang, "Efficient and secure multi-user multi-task computation offloading for mobile-edge computing in mobile IoT networks," *IEEE Transactions on Network and Service Management*, vol. 17, no. 4, pp. 2410–2422, 2020.

[25] S. Wang, A. Pathania, and T. Mitra, "Neural network inference on mobile SoCs," *IEEE Design & Test*, vol. 37, no. 5, pp. 50–57, 2020.

[26] E. Strubell, A. Ganesh, and A. McCallum, "Energy and policy considerations for deep learning in NLP," 2019, http://arxiv.org/abs/1906.02243.

[27] J. Liu, J. Liu, W. Du, and D. Li, "Performance analysis and characterization of training deep learning models on mobile device," in *IEEE 25th International Conference on Parallel and Distributed Systems (ICPADS)*, pp. 506–515, Tianjin, China, 2019.

[28] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: visual explanations from deep networks via gradient-based localization," in *IEEE international conference on computer vision*, pp. 618–626, Venice, Italy, 2017.

[29] T. A. Almeida, J. M. G. Hidalgo, and A. Yamakami, "Contributions to the study of SMS spam filtering: new collection and results," *11th ACM symposium on Document engineering*, pp. 259–262, 2011.

[30] The Apache Software Foundation, "SpamAssassin public mail corpus," 2006, https://spamassassin.apache.org/old/publiccorpus.

[31] S. Arzt, S. Rasthofer, C. Fritz et al., "Flowdroid: precise context, flow, field, object-sensitive and lifecycle-aware taint analysis for android apps," *ACM SIGPLAN Notices*, vol. 49, no. 6, pp. 259–269, 2014.

[32] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[33] Y. Kim, "Convolutional neural networks for sentence classification," in *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar, 2014.

[34] B. Chen, Z. Ren, C. Yu, I. Hussain, and J. Liu, "Adversarial examples for CNN-based malware detectors," *IEEE Access*, vol. 7, pp. 54360–54371, 2019.

Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 5883324, 12 pages
https://doi.org/10.1155/2022/5883324

Research Article

A Novel Image Edge Detection Method Based on the Asymmetric STDP Mechanism of the Visual Path

Tao Fang[1], **Jiantao Yuan**[1], **Rui Yin**[1] and **Celimuge Wu**[2]

[1]*School of Information & Electrical Engineering, Zhejiang University City College, Hangzhou 310015, China*
[2]*Graduate School of Informatics and Engineering, The University of Electro-Communications, Tokyo 182-8585, Japan*

Correspondence should be addressed to Rui Yin; yinrui@zucc.edu.cn

Received 28 March 2022; Accepted 7 May 2022; Published 7 June 2022

Academic Editor: Yuanlong Cao

The detection of image edges plays an important role for image processing. In view of the fact that these existing methods cannot effectively detect the edge of the image when facing the image with rich details. This paper proposes a novel method of asymmetric spike-timing-dependent plasticity (STDP) image edge detection based on the visual physiological mechanism. In the proposed method, the original image is preprocessed by the Gabor filter to simulate the visual physiological orientation characteristics to obtain the image information in different directions, and the orientation feature fusion is used to reconstruct the primary edge feature information of the image. Then, based on the mechanism of the visual nervous system, a neuron network composed of dynamic synapses based on the asymmetric STDP mechanism is constructed to further process it to obtain impulse response images. In order to eliminate disturbance of the neuron's system noise on the impulse response image, the impulse response image is filtered by a Gaussian filter. Then, the lateral inhibition between neurons is applied to refine the filtered image edges. Finally, the result is normalized, and the final edge of the experimental image is obtained. Experimental results based on the colony image data set collected in the laboratory indicate that the proposed method achieved better performance than these state-of-the-art methods; meanwhile, the AUC value remains above 0.6.

1. Introduction

At present, for the low-level visual feature processing of images based on bionic vision, many related theories have been proposed and good experimental results have been obtained [1–3]. The edges and contours are the dominant features to describe an image; hence, these two features are usually employed for higher-level image processing. Detection of edges and contours is a hot topic in the field of image processing. It is widely used in computer vision fields such as image classification, target detection, and image segmentation [4–7]. How to effectively and accurately detect the edges and contours of the image is of great significance to the subsequent image analysis, recognition, and understanding.

A number of methods have been proposed for image edge detection. This mainly includes the following: (1) the method based on the space domain, which is mainly based on spatial calculations, uses a relatively primitive differential operator, and judges the position of the edge based on the extreme value of the first derivative of edge gray value and the zero-crossing point of the second derivative. The template of the edge detection operator is convolved with the input image to directly perform edge detection on the acquired image. The Roberts operator [8] achieved high edge positioning accuracy, and the detected edges are relatively delicate, but it is sensitive to noise and of poor robustness. It is easy to cause local edge loss and cause the edge contour of the detected object to be discontinuous. Although the Sobel operator [9] and Prewitt operator [10] can suppress noise, the detection boundary line is wider. Compared with the above-mentioned method, the Canny operator [11] is relatively insensitive to noise but is susceptible to the influence of gradient amplitude and double thresholds and detects false edges and edge discontinuities. The Laplace operator [12] is of isotropy, linearity, and displacement invariance, but it needs to perform two-level difference processing while obtaining the edge, which produces a double-pixel edge and doubles the noise and affects the detection

accuracy. The Log operator [13] is implemented on the basis of the Laplace operator. First, the Gaussian function is used to low-pass filter the noise existing in the original image, and then, the Laplace operator is used for edge detection. Compared with the Laplace operator, although the noise in the image is suppressed, it weakens some low-intensity edges and causes a discontinuity in edge detection. (2) Another is a method based on the transform domain, which transforms the image to the corresponding transform domain through various image transformations, obtains the coefficient matrix, and performs a certain correction on the coefficient matrix to obtain the result. For example, wavelet transform [14] uses the transformed high-frequency components to eliminate the sudden change information and noise in the image. However, wavelet is not optimal in terms of the sparsity of the representation function, and the scale of wavelet transform is difficult to be unified, which will cause the contradiction between edge positioning accuracy and noise. Mathematical morphology [15] is a method that uses nonlinear filtering. By introducing the basic features and structure of the image, the problem of image processing such as noise suppression, feature extraction, and edge detection is solved; meanwhile, it balances off the detection accuracy and antinoise performance. However, there are shortcomings such as the problem of a single structural element and poor performance of edge detection in the context of rich details.

The increasements of the complexity and diversity of images require more effective edge detection methods. With the advancement of the physiological experiments of the visual mechanism in recent years, a large number of results have been obtained, which enables people to have a certain understanding of the cognitive process of vision. Given the near-perfect ability of the human visual system in processing complex image tasks, it can eliminate noise well and has extremely strong fault tolerance, which is unmatched by any existing image processing technology. Therefore, the human visual system currently provides inspiration and guidance while proposing novel models for image processing [16, 17]; e.g., image edge detection based on PCNN, which simulates the distribution and transmission of neuron pulse information flow, gives full play to the nonlinear modeling ability of neuron network in edge detection [18]. As well as the Gabor filter that simulates the direction selectivity of the visual nervous system, it has also been better applied in edge detection [19]. The paper [20] studied the experimental and theoretical methods for searching for effective local training rules for unsupervised pattern recognition through high-performance memristor spike neural networks. The paper [21] proposed a temporal preprocessing model of video frames using a biologically inspired vision model, and the bioinspired model consists of multiple layers of processing analogous to the photoreceptor cells in the visual system of small insects. There are also some deep learning-based methods for edge detection. For example, the paper [22] uses a spherical camera and two personal computers to build a remote apple growth monitoring hardware system and obtain apple images regularly. A fusion convolutional feature (FCF) edge detection network is designed to segment apple images for remote estimation of the apple size throughout the growth period. The paper [23] proposes an edge detection model with improved performance based on the convolutional neural networks and Laplacian filters, and the proposed method successfully detected the fuzzy defects on the noisy X-ray image. However, these methods lack an in-depth study of the related physiological mechanisms; moreover, the experimental objects are also homogeneous.

In order to address these issues in these existing methods, this paper studies the application of image edge detection based on neuron pulse emission coding under the asymmetric STDP mechanism of the visual pathway from the perspective of biological vision. Consider that the synapse is the key physiological structure for the effective transmission of impulse information between biological neurons [24]. It will be subject to changes in the intensity of the stimulation signal inside and outside of the biological organism, constantly self-adjust and change, and reshape the connection strength between neurons to meet the needs of biological nervous system information processing and action guidance. The asymmetric STDP information processing mechanism of the visual pathway of excitatory and inhibitory synapses is studied. This is in view of the fact that the neurotransmitters found in physiological anatomy experiments can be divided into two types: excitatory neurotransmitters and inhibitory neurotransmitters [25]. These different types of neurotransmitters also play a very important role in the process of biological visual information processing and play a decisive role in regulating the synaptic connections between neurons. Therefore, the time windows of long-term potentiation (LTP) and long-term inhibition (LTD) based on dynamic synaptic plasticity are asymmetric. This paper proposes that the Izhikevich neurons are used as the basic nodes of the network, and the information flow between neurons is transmitted through the physiological structure of dynamic synapses, and the information flow pulses are coded in time series. At the same time, the lateral inhibition mechanism of information transmission between neurons based on physiological experiments can be used to improve the contrast between the edge pixels of the image and the background information of the image. This can make the edges of the image richer and provide better basic feature information of the main content of the image for subsequent higher-level image-related tasks.

The rest of the present study is organized as follows. In Section 2, the experimental materials and methods of this article are introduced. In Section 3, experimental results are discussed and analyzed. The conclusion is drawn in Section 4.

2. Materials and Methods

In this paper, by simulating the visual processing mechanism, a dynamic synaptic neuron network based on the asymmetric STDP mechanism is constructed to realize the effective detection of image edges. According to physiological experiments, neurons in the visual cortex have direction selectivity for input stimuli. In view of the fact that the frequency and direction of the Gabor filter are similar to the human visual system, the

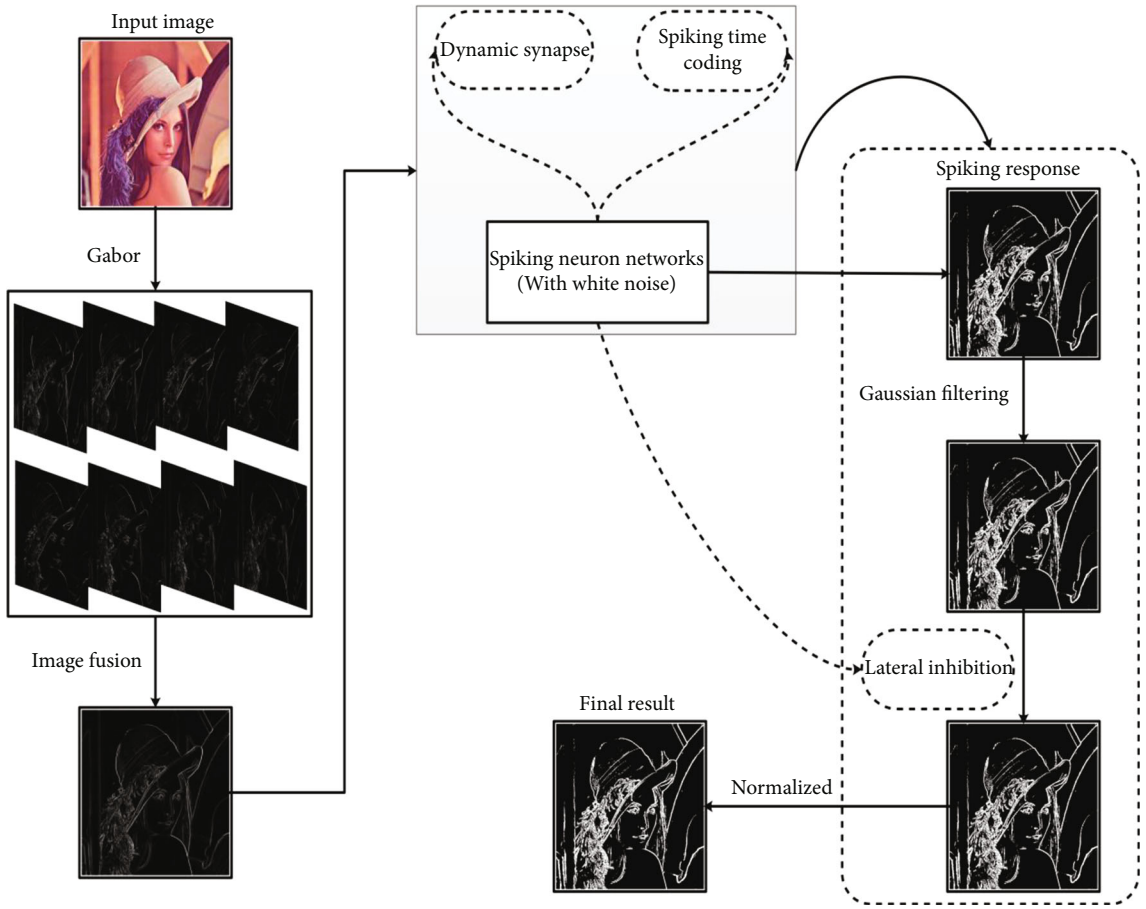


Figure 1: Algorithm block diagram.

original image is preprocessed by using the Gabor filter to simulate the human visual mechanism to obtain image features in different directions. After the feature fusion is carried out, it will be transmitted to a neuron network composed of dynamic synapses based on the asymmetric STDP mechanism. By recording the first pulse firing time of the neuron, the information flow processing scheme based on time sequence coding is given. As the neurons in the visual pathway are disturbed by a lot of noise, Gaussian filtering is performed on the pulse information stream based on timing coding. In addition, considering the physiological mechanism of lateral inhibition between neurons, this paper simulates the mechanism of lateral inhibition, further processes the image after Gaussian filtering, and finally obtains the image after neuron lateral inhibition. Finally, the normalization process is performed to obtain the final edge of the image. The specific process is shown in Figure 1.

According to related physiological research, this paper constructs a dynamic synaptic network with the Izhikevich neurons as the basic unit and uses time series coding for the information flow pulse [26]. At the same time, the important physiological significance of excitatory synapses/inhibitory synapses in the process of visual information processing and processing is considered, as well as visual physiological mechanisms such as asymmetric STDP mechanism and lateral inhibition (the specific structure is shown in Figure 2). Neurons promote or inhibit each other through the formation of excitatory synaptic transmitter AMPA/inhibitory synaptic transmitter GABA and dynamically adjust the weight of synaptic connections between neurons. And based on the asymmetric STDP mechanism, it realizes the effective transmission and processing of various sensory information [27]. In this article, the processing of visual information is mainly considered.

The neuron model is an important foundation of the neuron network. Taking into account the computational efficiency, complexity, and mathematical analysis performance of the existing neuron model. In this paper, the Izhikevich neuron model is used to construct a pulsed neuron network, and its mathematical model is shown in

$$\begin{cases} v_i' = 0.04v_i^2 + 5v_i + 140 - u_i + \gamma I_i + \sum_{j=1}^{N} w_{ij}\left(v_j - v_{eq}\right) + \xi_i(t), \\ u_i' = a(bv_i - u_i), \\ \text{if } v_i \geq 30,\ c \longleftarrow v_i,\ u_i + d \longleftarrow u_i, \end{cases} \tag{1}$$

where a, b, c, $d$are model parameters. $t$is the time variable. v_i is the membrane potential of neuron i. u_i is the recovery

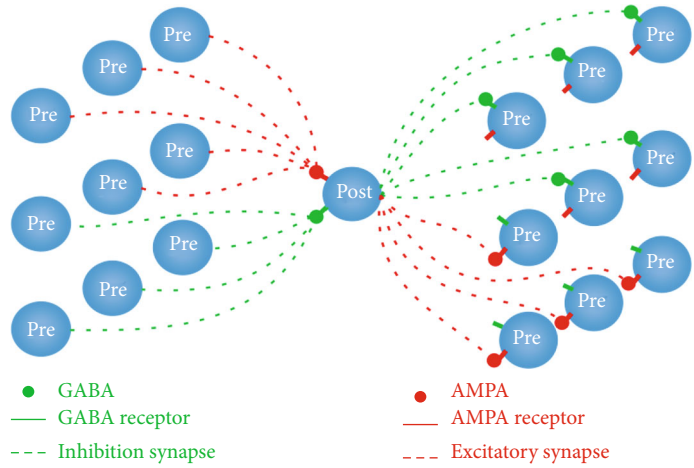


Figure 2: Dynamic synaptic network.

variable of neuron i. $\xi_i(t)$ is Gaussian white noise with intensity D. v_{eq} is the threshold value of the membrane voltage. w_{ij} is the strength of the synaptic connection from the j-th neuron to the i-th neuron. The external input is γI_i, where

$$I_i = x_i\theta(t), x_i \in \{0, 1\},$$
$$\theta(t) = \begin{cases} 1, t \geq 0, \\ 0, t < 0, \end{cases} \quad (2)$$

where x_i is a binary factor, which indicates whether the i-th neuron has input, and γ is the strength of the external input signal.

If the system parameters are inconsistent, the Izhikevich neuron model will show different firing patterns. In this paper, we take $a = 0.02$, $b = 0.2$, $c = -65$, and $d = 8$, and the intensity of white noise is $D = 0.01$.

2.1. Asymmetric STDP Mechanism. Synapse is the key structure for information transmission between neurons, and it is constantly changing and remodeling its connection strength to meet the needs of all aspects of the body due to changes in the body and outside of the body [28]. A complete synapse is composed of the presynaptic membrane, postsynaptic membrane, and synaptic cleft in between neurons. The presynaptic membrane has vesicles that store neurotransmitters, and there are receptors for the corresponding neurotransmitter on the postsynaptic membrane. After the presynaptic membrane is electrically or chemically stimulated, the vesicles release neurotransmitters to the synaptic cleft and bind to the corresponding receptors on the postsynaptic membrane to generate various electrical activities (local potentials). Then, it spreads to the corresponding neural circuits in a short period of time, producing different neurobehavioral activities [29].

The two main types of neurotransmitters in the brain are excitatory transmitters and inhibitory transmitters [30]. These transmitters also play an important role in the process of visual information processing. AMPA is the vast majority of excitatory synaptic transmitters in the brain. Synaptic plasticity, that is, the dynamic changes of neuron synaptic performance, is considered to be the basis of information encoding and storage in learning and memory. One of the most important mechanisms is that the regulation of synaptic strength is closely related to the regulation of AMPA receptor transport in the synapse [31]. GABA is the most widely distributed inhibitory neurotransmitter in the central nervous system. The specific mechanism of GABA is that GABA released from the presynaptic membrane binds to the GABA receptors of the postsynaptic membrane to form a receptor complex and undergo configuration changes, activate ion channels, allow ions to pass selectively, and cause neuronal hyperpolarized. It inhibits the excessive discharge of excitatory neurons and finally plays a role in hindering the transmission of nerve signals [32]. The existence of excitatory synapse/inhibitory synapse is an important part of information transmission in the nervous system, and it is also of great significance to synaptic plasticity.

Studies have found that the time sequence of presynaptic spikes and postsynaptic spikes affects the strength of the connections between presynaptic and presynaptic neurons. Hebb first interpreted this phenomenon from a mathematical point of view and proposed the Hebb learning rule. The principle is to increase or decrease the connection weight of the synapse according to the correlation of the firing of the neurons before and after the synapse.

According to the experimental results [33], the relationship between the time difference between the two neurons to produce nerve impulses on the excitatory post-synaptic current (EPSC). Aiming at the time asymmetry of synaptic plasticity changes, Bi and Poo further proposed the "spike-timing-dependent plasticity" mechanism. According to the length of the time course, it can be divided into short-term plasticity and long-term plasticity (mainly including LTP and LTD). They can unsupervised and autonomously adjust the synaptic weights of neural networks, more accurately describe the changes in the weight connections of neurons in biology, and amend the Hebb learning rule.

The LTP/LTD change time window of asymmetric STDP synaptic plasticity is asymmetric, and its essence is based on the interval of neuron firing time, reflecting the

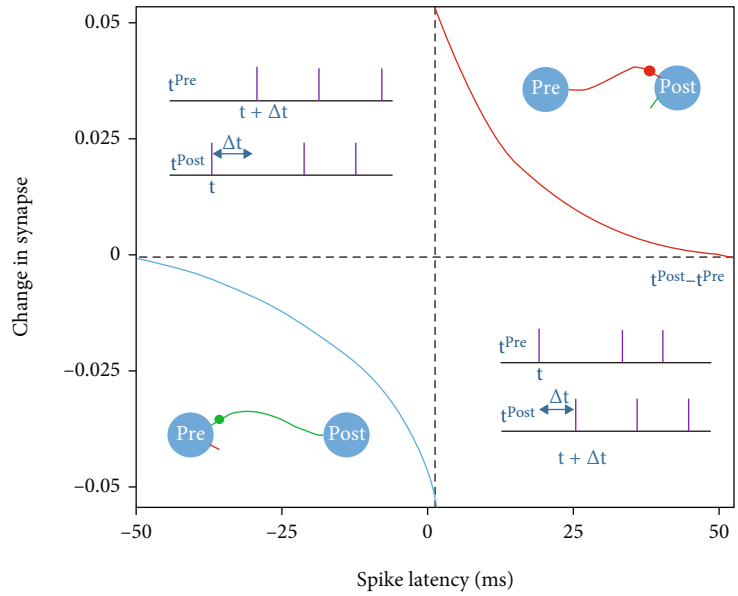


Figure 3: Schematic diagram of asymmetric STDP mechanism.

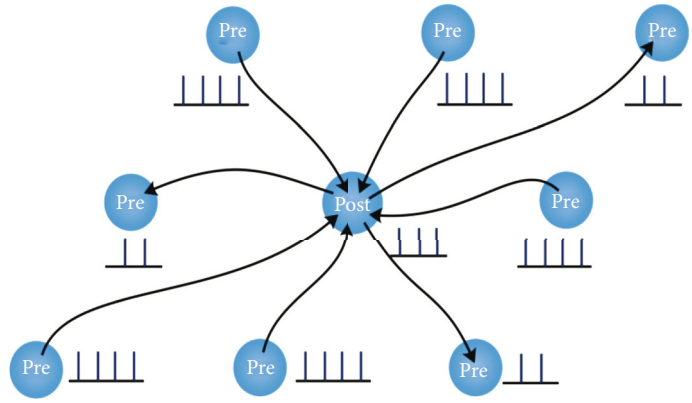


Figure 4: Principle of lateral suppression.

causal relationship between neurons in the form of directional connections. The typical asymmetric STDP learning mechanism is shown in Figure 3. When the presynaptic neuron pulse firing time t^{Pre} is before the postsynaptic neuron pulse firing time t^{Post}, that is, $t^{\mathrm{Pre}} < t^{\mathrm{Post}}$, the strength of the synaptic connection between them will increase; on the contrary, for $t^{\mathrm{Pre}} \geq t^{\mathrm{Post}}$, the strength of synaptic connections is weakened. Its function expression is shown in

$$\Delta g_{ji} = g_{ji} S(\Delta t),$$
$$S(\Delta t) = \begin{cases} A_+ * \exp\left(-\Delta t/\tau_+\right) \text{ if } \Delta t \geq 0, \\ A_- * \exp\left(\Delta t/\tau_-\right) \text{ if } \Delta t < 0, \end{cases} \quad (3)$$

where g_{ji} represents the connection strength between neuron i and neuron j. Δt is the difference between the time when the presynaptic cell produces spike and the time when the postsynaptic neuron produces spike, that is, $\Delta t = t_j - t_i$. $S(\Delta t)$ is the STDP adjustment function. The parameters A_+ and A_- affect the adjustment range. The larger their value is, the larger the synaptic connection strength increases or decreases within one step. τ_+ and τ_- are the delay constants of STDP adjustment parameters, and $\tau_+ = 25$, $\tau_- = 15$, $A_+ = 0.05$, and $A_- = 1.05 * A_+ = 0.0525$ are used in this article.

2.2. Lateral Inhibition. Hartline discovered the phenomenon of lateral inhibition for the first time when conducting monocular electrophysiological experiments on Limulus. According to further in-depth experiments on visual physiology, it is found that visual lateral inhibition is carried out in the analog signal part, which has an important manifestation in horizontal cells. When horizontal cells receive a signal from a light information pathway, they are affected by glutamate released by photoreceptor cells and release GABA, which inhibits the release of glutamate from receptor cells in other light information pathways, thereby weakening the response of adjacent pathways.

According to this physiological phenomenon, this article introduces the lateral inhibition between neurons in the cerebral cortex when constructing the interconnection of neuronal networks. When a neuron is excited, it will inhibit

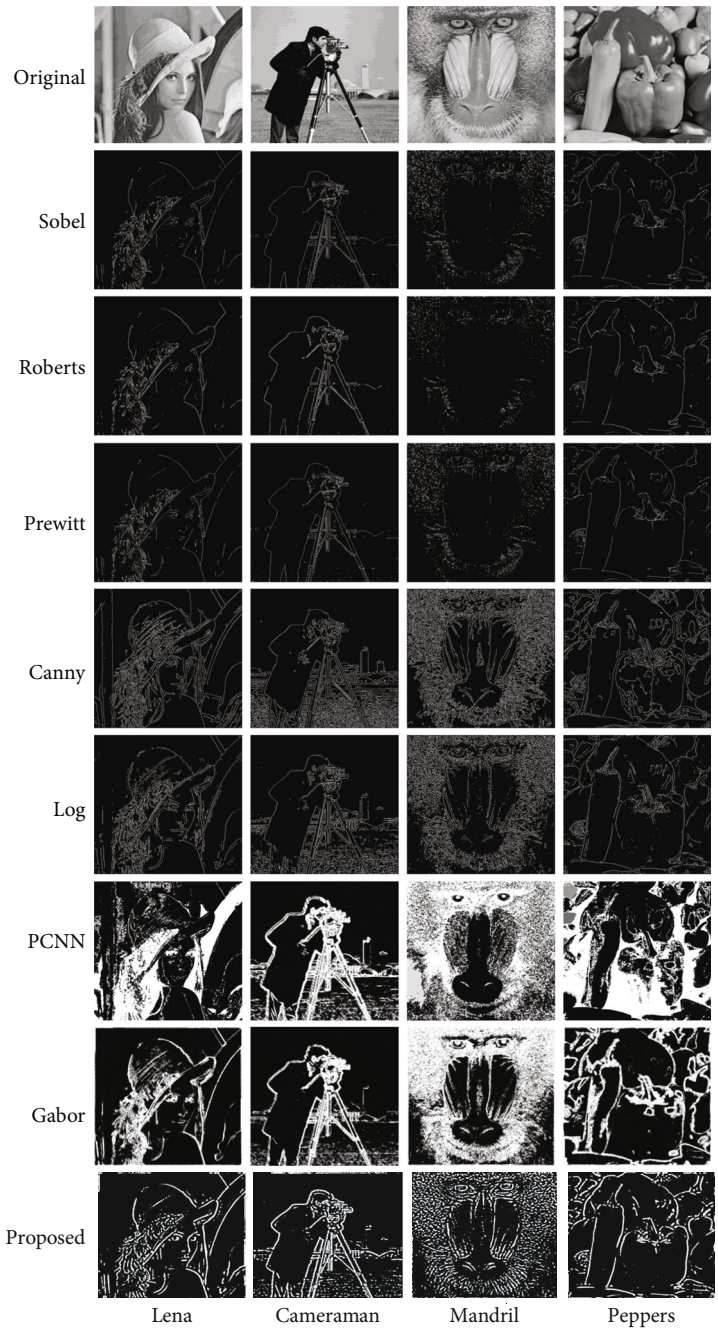


Figure 5: Edge detection result.

Table 1: The AUC value of the edge image under various experimental algorithms.

Image	Method							
	Sobel	Roberts	Prewitt	Canny	Log	PCNN	Gabor	Proposed
Lena	0.6232	0.6320	0.5735	0.6256	0.6093	0.6393	0.6683	0.8863
Cameraman	0.6033	0.6227	0.5842	0.6084	0.6320	0.5030	0.5993	0.8049
Mandril	0.5534	0.5546	0.5432	0.5513	0.5615	0.5133	0.5406	0.7519
Peppers	0.5802	0.6019	0.5678	0.5860	0.6140	0.6011	0.6076	0.7949
FPS	28	21	19	14	16	1/4	1	2

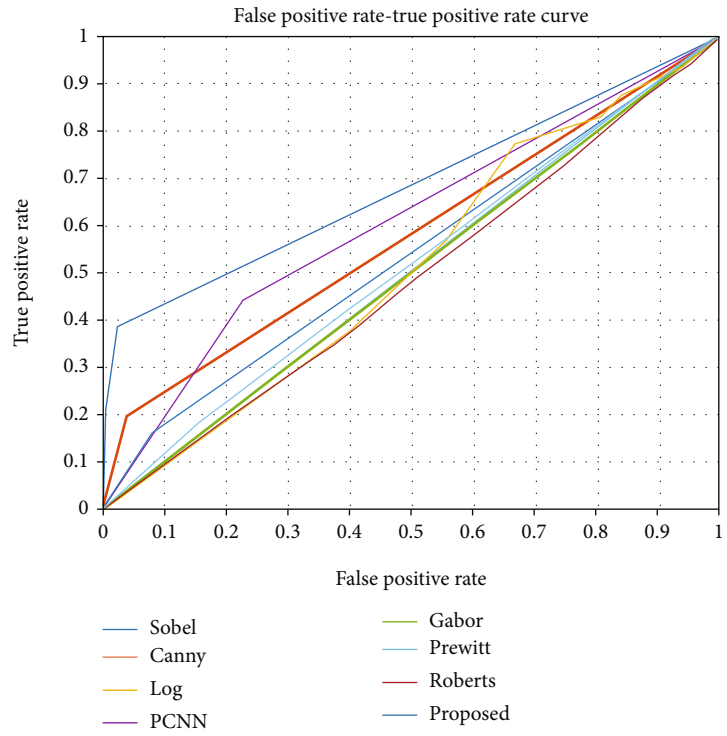


Figure 6: ROC curves of different algorithms.

the excitement of other neurons in the area, which can significantly enhance the recognition of the visual system during target recognition and enhance the recognition of image edge information. This article considers the sequence of neuron firing and adopts the neuron lateral inhibition method as shown in Figure 4. If the neuron at the center of the receptive field is excited earlier than other neurons, it will have an inhibitory effect on the corresponding neuron. If the peripheral neurons are excited first, the central neurons will be inhibited. In this way, not only the edge points of the image can be highlighted, but also different levels of information can be processed in a targeted manner, so as to express rich image edge information.

To reduce complexity and facilitate calculations, a 3×3 receptive field window is constructed, and the lateral inhibition of adjacent neurons is shown in

$$\vartheta'_{\text{Post}} = \begin{cases} \vartheta_{\text{Post}} \exp\left(-\dfrac{\vartheta_{\text{Post}}}{\vartheta_{\text{Pre}}}\right) & (\vartheta_{\text{Post}} < \vartheta_{\text{Pre}}), \\ \vartheta_{\text{Post}} \exp\left(\dfrac{\vartheta_{\text{Pre}}}{\vartheta_{\text{Post}}}\right) & (\vartheta_{\text{Post}} > \vartheta_{\text{Pre}}), \\ \vartheta_{\text{Post}} \, (\vartheta_{\text{Post}} = \vartheta_{\text{Pre}}), \end{cases} \tag{4}$$

$$\vartheta'_{\text{Pre}} = \begin{cases} \vartheta_{\text{Pre}} \exp\left(-\dfrac{\vartheta_{\text{Pre}}}{\vartheta_{\text{Post}}}\right) (\vartheta_{\text{Pre}} < \vartheta_{\text{Post}}), \\ \vartheta_{\text{Pre}} \exp\left(\dfrac{\vartheta_{\text{Post}}}{\vartheta_{\text{Pre}}}\right) (\vartheta_{\text{Pre}} > \vartheta_{\text{Post}}), \\ \vartheta_{\text{Pre}} (\vartheta_{\text{Pre}} = \vartheta_{\text{Post}}), \end{cases} \tag{5}$$

where ϑ_{Post}, ϑ'_{Post}, respectively, represent the central element before and after the update in the receptive field window and ϑ_{Pre}, ϑ'_{Pre}, respectively, represent the noncentral element before and after the update in the receptive field window.

3. Experimental Results and Discussion

In order to verify the effectiveness of the method in this paper, the image data sets such as Lena, which are commonly used in edge detection, and the colony image data sets collected by this research group in the laboratory are used as the experimental objects. Among them, the colony map is obtained by the laboratory using the imitating natural light suspension dark-field system, through the F/1.4 large aperture lens, and the colony after the Petri dish culture is obtained by imaging at the level of tens of millions of pixels. This article installs Matlab R2016b version on Ubuntu 20.04 LTS version for experiment. The main hardware includes AMD Ryzen5 5600H CPU and 16GB memory.

And compare the results of the method proposed in this article with traditional edge detection methods such as the Sobel, Roberts, Prewitt, Canny, Log, PCNN, and Gabor; the experimental results are shown in Figure 5. Among them, the first row is the original picture. The second row is the test result of the Sobel method. The third row is the test result of the Roberts method. The fourth row is the detection result of the Prewitt method. The fifth row is the result of the Canny method. The sixth row is the test result of the Log method. The seventh row is the PCNN method. The eighth row is the test result of the Gabor method. The ninth row is the test result of the method in this paper. From the experimental results in

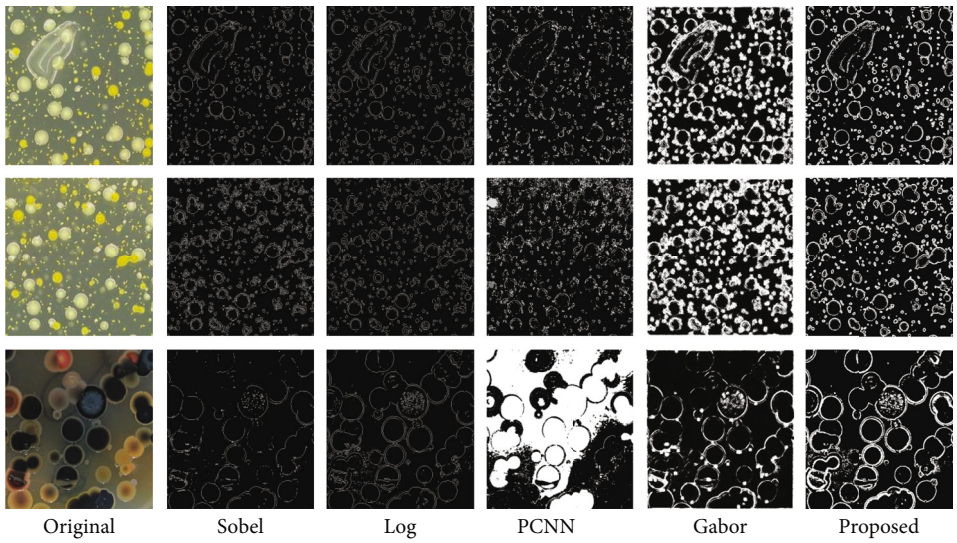


Figure 7: The effect of colony edge experiment. The first column is the original image; the second column is the detection result of the Sobel method; the third column is the detection result of the Log method; the fourth column is the detection result of the PCNN method; the fifth column is the detection result of the Gabor method; the sixth column is the result of this method.

Table 2: Information entropy value of edge image under various detection algorithms.

Image	Method				
	Sobel	Log	PCNN	Gabor	Proposed
Colony 1	0.6010	0.6892	0.5523	0.7185	0.7309
Colony 2	0.5801	0.6217	0.6168	0.5800	0.7763
Colony 3	0.9652	0.9857	0.9792	0.8540	0.9862

Figure 5, it can be seen that the Sobel and other methods can detect the more obvious edge information of the salient target when targeting simpler pictures such as Peppers but will lose most of the edge information of the background. In addition, for images with richer details such as Lena, although the overall edge information of the image can be outlined, most of the detailed edge information will also be lost, resulting in discontinuous edge detection.

In Figure 5, it can be intuitively found that although the detected edge information of Log is richer, it has certain shortcomings. It will cause excessive segmentation during detection, which is too sensitive to noise points, and the detected image edge information is too redundant. This affects the subsequent processing of the image and is not conducive to observation. From the experimental results, it can be found that the edge continuity detected by the algorithm in this paper is good, and a better single-pixel edge can be obtained after refinement. The edge detection accuracy is high, and the edge information can be highlighted, for example, for images such as Lena. In the case of preventing excessive segmentation, their detailed edge information can still be well characterized, and it has a better detection effect than other edge detection methods.

The 1st row is the original image. The 2nd row is the detection result of the Sobel method. The 3rd row is the result of the Roberts method. The 4th row is the detection result of the Prewitt method. The 5th row is the test result of the Canny method. The 6th row is the detection result of the Log method. The 7th row is the detection result of the PCNN method. The 8th row is the detection result of the Gabor method. The 9th row is the test result of the method in this paper. On the basis of qualitative analysis, in order to better quantitatively compare the experimental results of different methods, this article uses the ROC/AUC indicators commonly used in machine learning to evaluate the experimental effects of various edge detection methods. The ROC curve mainly includes two indicator parameters: false positive rate (FPR) and true positive rate (TPR). A series of target values can be obtained by changing the threshold, and the ROC curve can be drawn with TPR as the ordinate and FPR as the abscissa. AUC is the sum of the ROC curve and the accumulated area under the horizontal axis. The larger the area, the better the edge detection effect of the image, and vice versa, the poorer the edge detection effect. The specific calculation is shown in (6). In Figure 5, it can be intuitively found that although the detected edge information of Log is richer, it has certain shortcomings. It will cause excessive segmentation during detection, which is too sensitive to noise points, and the detected image edge information is too redundant. This affects the subsequent processing of the image and is not conducive to observation. From the experimental results, it can be found that the edge continuity detected by the algorithm in this paper is good, and a better single-pixel edge can be obtained after refinement. The edge detection accuracy is high, and the edge information can be highlighted, for example, for images such as Lena. In the case of preventing excessive segmentation, their detailed edge information can still be well characterized, and it has a better detection effect than other edge detection methods.

$$\begin{cases} \mathrm{TPR} = \dfrac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}, \\ \mathrm{FPR} = \dfrac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}}, \end{cases} \tag{6}$$

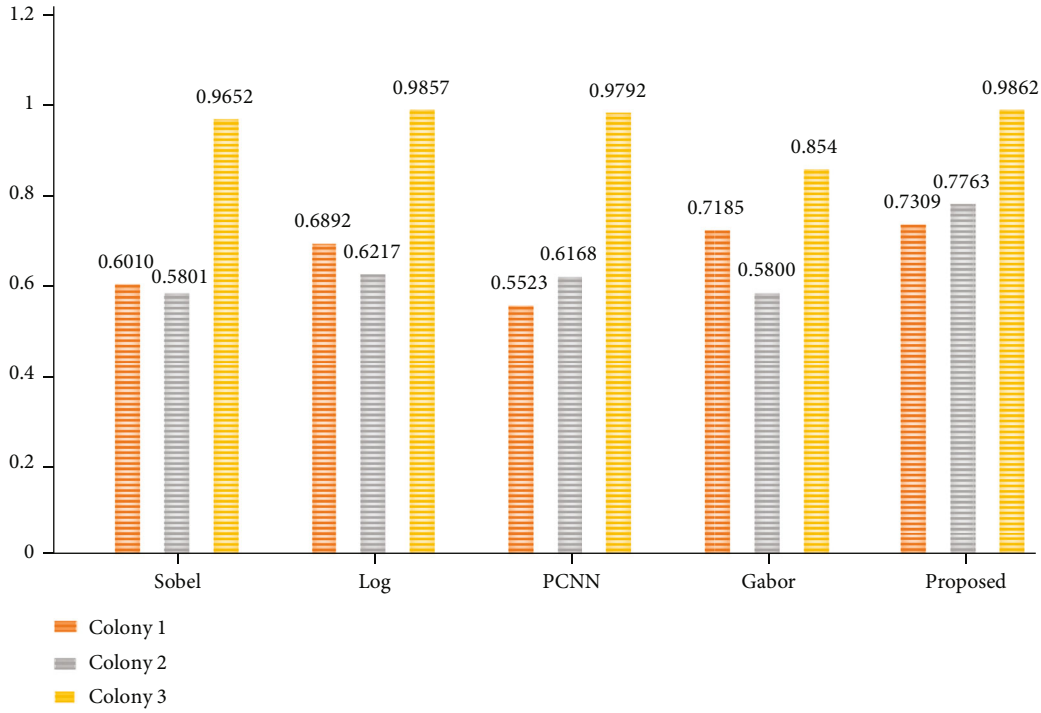


Figure 8: Visualization of information entropy of experimental results of different algorithms.

where TP represents the pixel set that correctly classifies the positive example as the positive example under different thresholds. TN represents the pixel set that correctly classifies negative examples as negative examples under different thresholds. FP represents the set of pixels that misclassify negative examples as positive examples under different thresholds. FN represents a set of pixels that incorrectly classify a positive example as a negative example under different thresholds.

In the process of calculating AUC, it is necessary to obtain the ground truth of the image. Since manual handdrawing is inefficient and subjective, an objective and automatic judgment method is needed to obtain the reference map. In this paper, N edge images are obtained by taking multiple thresholds for different edge detection methods, and the number of edge points at the same position in the N edge images is counted, so as to obtain N candidate edge images. Then, the best candidate edge image is determined by combining ROC statistical indicators and the diagnosis line, and this image is used as the edge reference image. Then, according to the above-mentioned index calculation method, this paper, respectively, calculated the Sobel, Roberts, Prewitt, Canny, Log, and the ROC/AUC index value of this method relative to the edge reference image. The AUC values of this method in Lena, Cameraman, Mandril, and Peppers can reach 0.8863, 0.8049, 0.7519, and 0.7949, respectively, which are significantly higher than the comparative experimental method, indicating the effectiveness of the edge detection method in this paper. The specific results are shown in Table 1. It can be intuitively found from Figure 6 that the ROC curve of the method proposed in this paper is closer to the upper left corner of the coordinate axis, indicating that the method proposed in this paper is superior to the existing traditional methods in the effect of image edge detection.

At the same time, this article counts the processing speed of different algorithms in image detection. According to the experimental results, it can be found that the edge detection speed of the Sobel, Log, and Gabor images based on traditional mathematical methods is relatively faster, but their edge detection accuracy is lower. Compared with the PCNN algorithm, which is also based on biological inspiration, the method in this paper has a faster processing speed while maintaining a higher edge detection accuracy.

At the same time, in order to further illustrate the effectiveness of this method, in this paper, the experimental objects are further processed, and the experimental results are further analyzed by using information entropy as the evaluation index. The larger the entropy value, the more edge detail information contained in the research object. The calculation is shown in

$$H = \sum_{i=0}^{255} p_i \log (p_i), \tag{7}$$

where i represents the gray value of the image and p_i represents the probability of the pixel with the gray value i in the image appearing in the image.

The experimental results are shown in Figure 7. By comparing the experimental images with rich details such as the colony, it can be intuitively found that the edge detection of the colony in this article has more obvious contrast and the continuity of the colony is more perfect. At the same time, it

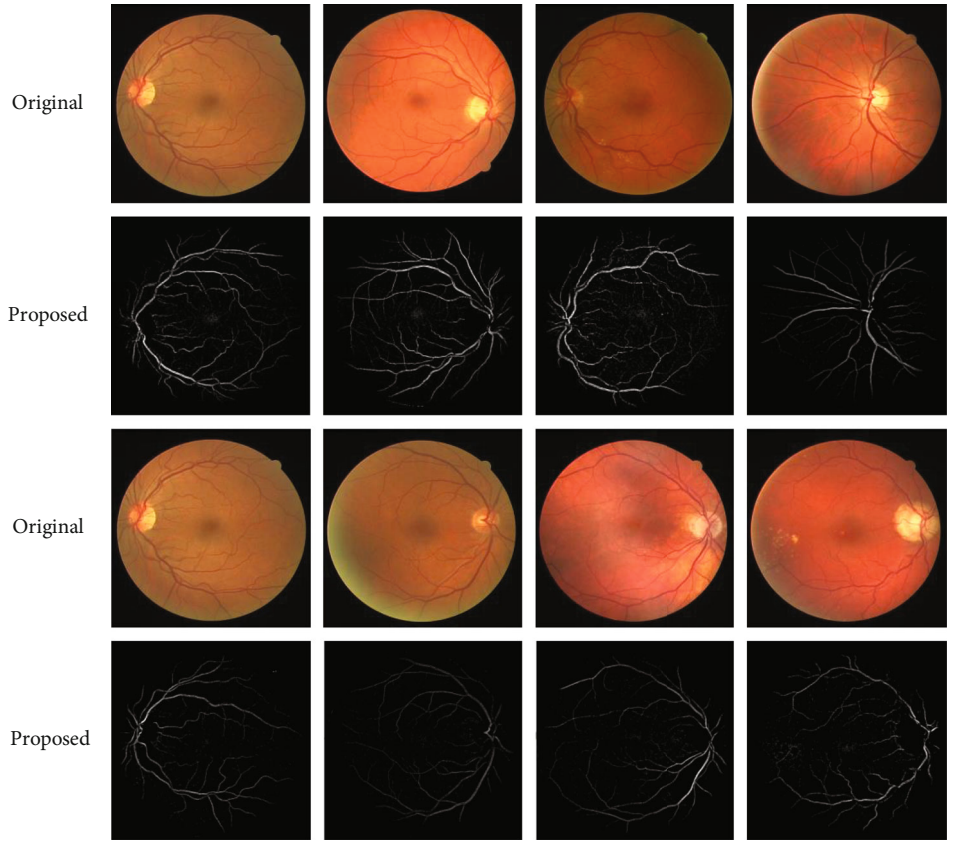


Figure 9: The result of segmentation and extraction of fundus blood vessels. The first and third rows are the original images of the fundus blood vessels, and the second and fourth rows are the segmentation results.

also has a very good detection effect under a small detection target. In addition, through formula (7), the method in this paper and several comparative experimental methods are used to calculate the information entropy of the edge detection results. The specific results are shown in Table 2. According to the quantitative analysis, the information entropy value of the method in this paper has a relatively large advantage over the methods such as the Sobel, Log, PCNN, and Gabor, reaching 0.7309, 0.7763, and 0.9862, respectively.

Through the visualization in Figure 8, it is obvious that the method in this paper can achieve higher information entropy in general and has greater advantages. This also shows that the method in this paper can obtain richer image edge details.

In addition, in order to further verify the effectiveness and scalability of the method proposed in this paper, this paper applies it to the segmentation and extraction of fundus blood vessels while keeping the parameters unchanged. Through the experimental results, it can be found that for different types of fundus blood vessels, the method in this paper can better extract the main blood vessel segmentation results, which also provides a good foundation for subsequent blood vessel processing tasks. The specific experimental results are shown in Figure 9.

Through the above qualitative analysis of the experimental results of different experimental methods, it can be found that the method in this paper can ensure the completeness and coherence of edge detection as much as possible while preventing the oversegmentation of the image and at the same time highlight the edge details of the image. This is of great significance for edge detection applied on images with richer details such as colonies. It is also necessary to improve the performance of this method through further research.

In addition, through further quantitative analysis of the experimental results, the AUC value of this method on different experimental images is relatively better than the existing traditional methods, which shows that the accuracy of edge detection is higher. The statistical analysis of the information entropy of the experimental results also shows that the method in this paper can retain more edge detail information when performing image edge detection. According to the qualitative and quantitative analysis of the experimental results, it consistently shows that the method in this paper has greater advantages over the existing traditional methods.

4. Conclusion

Different from traditional image edge detection methods based on spatial and exchange domains, this paper introduces asymmetric STDP, excitatory synapses/inhibitory synapses, time coding, and lateral inhibition based on physiological experiments related to visual physiological mechanisms. Through the introduction of its principle and function, the corresponding calculation model is established, and a method based on asymmetric STDP image edge detection is proposed. At present, the method proposed in this paper is mainly used

in the extraction of low-level visual features of the image and has a good effect in the edge detection of the colony image collected in the laboratory. The later application of this method in the field of image preprocessing has certain practical significance and at the same time provides some ideas for image processing methods based on vision mechanisms. How to further explain and simulate the characteristics of biological visual pathways will also be the focus of our next research work.

Data Availability

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

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the Ministry of Education Industry-University Cooperation Collaborative Education Project (202102019039), Zhejiang University City College Scientific Research Cultivation Fund Project (J-202223), ROIS NII Open Collaborative Research (22S0601), and JSPS KAKENHI (grant numbers 20H00592 and 21H03424).

References

[1] K. Yang, X. Zhang, and Y. Li, "A biological vision inspired framework for image enhancement in poor visibility conditions," *IEEE Transactions on Image Processing*, vol. 29, pp. 1493–1506, 2019.

[2] J. Wang, Y. Li, and K. Yang, "Retinal fundus image enhancement with image decomposition and visual adaptation," *Computers in Biology and Medicine*, vol. 128, p. 104116, 2020.

[3] X. Wu, H. Zhang, X. Hu, M. Shakeri, C. Fan, and J. Ting, "HDR reconstruction based on the polarization camera," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5113–5119, 2020.

[4] C. Rasche, "Rapid contour detection for image classification," *IET Image Processing*, vol. 12, no. 4, pp. 532–538, 2018.

[5] Y. Li, X. Sun, H. Wang, H. Sun, and X. Li, "Automatic target detection in high-resolution remote sensing images using a contour-based spatial model," *IEEE Geoscience and Remote Sensing Letters*, vol. 9, no. 5, pp. 886–890, 2012.

[6] L. Wang, Y. Chang, H. Wang, Z. Wu, J. Pu, and X. Yang, "An active contour model based on local fitted images for image segmentation," *Information Sciences*, vol. 418-419, pp. 61–73, 2017.

[7] T. Fang, Y. Fan, and W. Wu, "Salient contour detection on the basis of the mechanism of bilateral asymmetric receptive fields," *Signal, Image and Video Processing*, vol. 14, no. 7, pp. 1461–1469, 2020.

[8] H. Gong and L. Hao, "Roberts edge detection algorithm based on GPU," *Journal of Chemical and Pharmaceutical Research*, vol. 6, no. 7, pp. 1308–1314, 2014.

[9] N. Mathur, S. Mathur, and D. Mathur, "A novel approach to improve Sobel edge detector," *Procedia Computer Science*, vol. 93, pp. 431–438, 2016.

[10] R. Zhou, H. Yu, Y. Cheng, and F. Li, "Quantum image edge extraction based on improved Prewitt operator," *Quantum Information Processing*, vol. 18, no. 9, pp. 1–24, 2019.

[11] K. Gaurav and U. Ghanekar, "Image steganography based on Canny edge detection, dilation operator and hybrid coding," *Journal of Information Security and Applications*, vol. 41, pp. 41–51, 2018.

[12] C. Guo, M. Xiao, M. Minkov, Y. Shi, and S. Fan, "Photonic crystal slab Laplace operator for image differentiation," *Optica*, vol. 5, no. 3, pp. 251–256, 2018.

[13] S. Ghosal, J. Mandal, and R. Sarkar, "High payload image steganography based on Laplacian of Gaussian (Log) edge detector," *Multimedia Tools and Applications*, vol. 77, no. 23, pp. 30403–30418, 2018.

[14] A. Miri and K. Faez, "An image steganography method based on integer wavelet transform," *Multimedia Tools and Applications*, vol. 77, no. 11, pp. 13133–13144, 2018.

[15] L. Carazas and P. Sussner, "Detecao de Bordas baseada em Morfologia Matemática Fuzzy Intervalar e as Funcoes de Agregacao K," *Selecciones Matemáticas*, vol. 6, no. 2, pp. 238–247, 2019.

[16] D. Koniar, L. Hargaš, Z. Loncova, A. Simonova, F. Duchoň, and P. Beňo, "Visual system-based object tracking using image segmentation for biomedical applications," *Electrical Engineering*, vol. 99, no. 4, pp. 1349–1366, 2017.

[17] V. Prasath, D. Thanh, N. San, and S. Dvoenko, "Human visual system consistent model for wireless capsule endoscopy image enhancement and applications," *Pattern Recognition and Image Analysis*, vol. 30, no. 3, pp. 280–287, 2020.

[18] B. Cheng, L. Jin, and G. Li, "Infrared and visual image fusion using LNSST and an adaptive dual-channel PCNN with triple-linking strength," *Neurocomputing*, vol. 310, pp. 135–147, 2018.

[19] Y. Xiang, F. Wang, L. Wan, and H. You, "An advanced multiscale edge detector based on Gabor filters for SAR imagery," *IEEE Geoscience and Remote Sensing Letters*, vol. 14, no. 9, pp. 1522–1526, 2017.

[20] V. Demin, D. Nekhaev, I. Surazhevsky et al., "Necessary conditions for STDP-based pattern recognition learning in a memristive spiking neural network," *Neural Networks*, vol. 134, pp. 64–75, 2021.

[21] M. Uzair, R. Brinkworth, and A. Finn, "Bio-inspired video enhancement for small moving target detection," *IEEE Transactions on Image Processing*, vol. 30, pp. 1232–1244, 2021.

[22] D. Wang, C. Li, H. Song, H. Xiong, C. Liu, and D. He, "Deep learning approach for apple edge detection to remotely monitor apple growth in orchards," *IEEE Access*, vol. 8, pp. 26911–26925, 2020.

[23] A. Zx, A. Kys, and B. Mmg, "Development of a CNN edge detection model of noised X-ray images for enhanced performance of non-destructive testing," *Measurement*, vol. 174, p. 109012, 2021.

[24] C. Pedroarena, "A slow short-term depression at Purkinje to deep cerebellar nuclear neuron synapses supports gain-control and linear encoding over second-long time windows," *The Journal of Neuroscience*, vol. 40, no. 31, pp. 5937–5953, 2020.

[25] S. Sears and S. Hewett, "Influence of glutamate and GABA transport on brain excitatory/inhibitory balance," *Experimental Biology and Medicine*, vol. 246, no. 9, pp. 1069–1083, 2021.

[26] E. Izhikevich, "Simple model of spiking neurons," *IEEE Transactions on Neural Networks*, vol. 14, no. 6, pp. 1569–1572, 2003.

[27] J. Torres, F. Baroni, R. Latorre, and P. Varona, "Temporal discrimination from the interaction between dynamic synapses and intrinsic subthreshold oscillations," *Neurocomputing*, vol. 417, pp. 543–557, 2020.

[28] K. Rajagopal, S. Jafari, C. Li, A. Karthikeyan, and P. Duraisamy, "Suppressing spiral waves in a lattice array of coupled neurons using delayed asymmetric synapse coupling," *Chaos Solitons and Fractals*, vol. 146, p. 110855, 2021.

[29] S. Keene, C. Lubrano, S. Kazemzadeh et al., "A biohybrid synapse with neurotransmitter-mediated plasticity," *Nature Materials*, vol. 19, no. 9, pp. 969–973, 2020.

[30] P. Allen, I. Sommer, R. Jardri, M. Eysenck, and K. Hugdahl, "Extrinsic and default mode networks in psychiatric conditions: relationship to excitatory-inhibitory transmitter balance and early trauma," *Neuroscience and Biobehavioral Reviews*, vol. 99, pp. 90–100, 2019.

[31] N. Burnashev, A. Khodorova, P. Jonas et al., "Calcium-permeable AMPA-kainate receptors in fusiform cerebellar glial cells," *Science*, vol. 256, no. 5063, pp. 1566–1570, 2019.

[32] J. Storm-Mathisen, A. Leknes, A. Bore et al., "First visualization of glutamate and GABA in neurones by immunocytochemistry," *Nature*, vol. 301, no. 5900, pp. 517–520, 2019.

[33] G. Bi and M. Poo, "Synaptic modifications in cultured hippocampal neurons: dependence on spike timing, synaptic strength, and postsynaptic cell type," *Journal of Neuroscience*, vol. 18, no. 24, pp. 10464–10472, 1998.

Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 4282859, 19 pages
https://doi.org/10.1155/2022/4282859

Research Article

G/M/1-Based DDoS Attack Mitigation in 5G Ultradense Cellular Networks

Qinghang Gao,[1] **Hao Wang**,[1] **Liyong Wan**,[2] **Jianmao Xiao**,[1] **and Long Wang**[1]

[1]*School of Software, Jiangxi Normal University, Nanchang 330022, China*
[2]*Management Science and Engineering Research Center, Jiangxi Normal University, Nanchang 330022, China*

Correspondence should be addressed to Hao Wang; wanghao@jxnu.edu.cn

Received 24 January 2022; Revised 11 March 2022; Accepted 24 March 2022; Published 20 April 2022

Academic Editor: Alessandro Bazzi

With the 5G millimeter wave (mmWave) application, ultradense cellular networks are gradually becoming one of the core characteristics of 5G cellular networks. In the edge computing environment, considering load balancing among edge nodes is beneficial to slow down the process of distributed denial of service (DDoS) attack. However, most existing studies have given less consideration to congestion in the multiuser and multiedge server models. Someone who uses the M/M/1 model also seems to ignore the effect of scheduling algorithms on the Markov property of the task arrival process. In this manuscript, based on ensuring the quality of experience (QoE) for users, the G/M/1 model is introduced to the task scheduling of edge servers for the first time to improve load balancing between edge servers. For the multi armed bandit (MAB) algorithm framework, specific metrics are established to quantify the degree of its equilibrium. The number of users assigned to the edge nodes and each edge node's processing of specific tasks is taken into account. We experimentally evaluated its performance against two baseline approaches and three state-of-the-art approaches on a real-world dataset. And the experimental results validate the effectiveness of this method.

1. Introduction

As is known to all, user equipment (UE) has low computing capacity. It may not efficiently solve task requests initiated by users, while cloud services have problems such as long transmission delays. The presence of mobile edge computing (MEC) brings mobile computing, network storage, and control issues down from the cloud to the network edge, driving the execution of compute-intensive, latency-critical applications on mobile devices, effectively reducing latency and energy consumption [1–3].

There are still deficiencies in interoperability, heterogeneous architecture, data privacy, and load balancing in heterogeneous edge computing systems, which can be considered to be compensated for by requirements such as federated deployment and resource management [4]. Edge servers have limited memory, central processing unit (CPU), storage, and other resources. They generally deploy at base stations close to user terminals, and users are guaranteed low latency and stable connectivity by using edge servers [5]. The emergence of the user plane function (UPF) separates the control plane from the user plane, making MEC even more critical in 5G technology. The emergence of the 5G millimeter wave has significantly expanded the transmission bandwidth and reduced the transmission delay of mobile communications, but there are also challenges such as easy loss. Increasing the density of base stations (BSs) helps to minimize losses, and thus, ultradense cellular networks are gradually becoming one of the core features of 5G cellular networks [6]. The deployment of large-area and high-density BSs will bring new network security issues. Due to the limited signal transmission range and edge server resources, a typical IoT-based distributed denial of service (DDoS) attack can disable most nodes in a particular

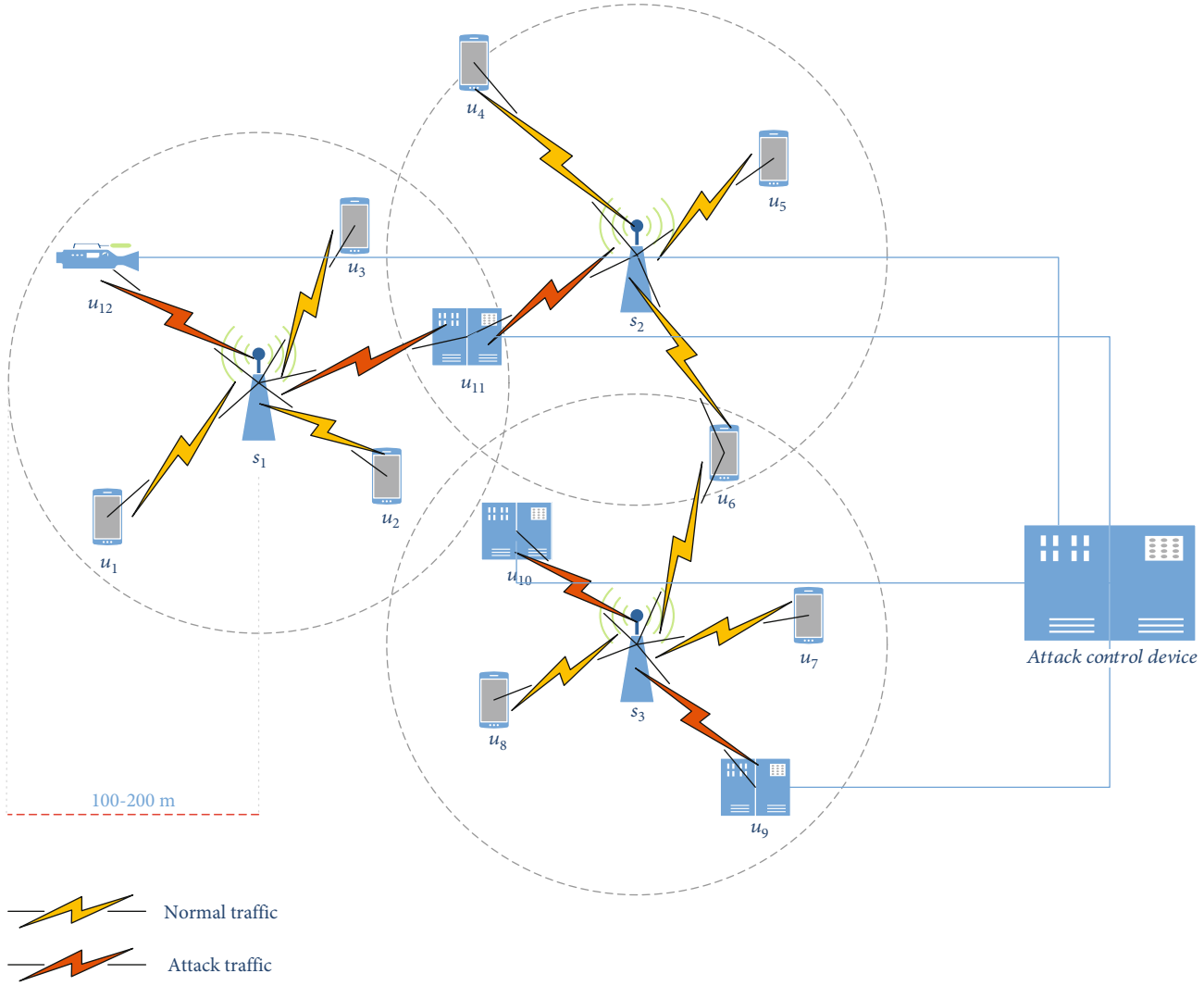


Figure 1: Ultradense cellular network topology diagram.

area by continuously trying to occupy the resources of edge nodes [7], thus paralyzing the Internet of Things (IoT) devices in its service interval within a specific period (smart monitors, infrared sensors, etc.) [8], causing severe social impacts and security problems.

DDoS attack is a resource competition problem between attackers and defenders [9], and this competition will be more prominent in resource-limited edge service environments [8]. Based on the careful consideration of system characteristics such as proximity constraint, capacity constraint, and delay constraint among edge servers [10], balancing the workload among edge servers can slow down the DDoS attack process [8], thus leaving enough reaction time for the system and reducing the possibility of the system being breached. We consider the load balancing problem of edge user allocation (EUA) based on users' quality of experience (QoE) and establish specific metrics to quantify the degree of balance. To more precisely quantify the QoE, we introduce the multi-armed bandit (MAB) algorithm framework and add nonstationary factors to the learning mechanism to better adapt to the actual complex and variable task assignment process.

For the time being, there is relatively little research on the load balancing problem and mainly reflected in the relatively balanced number of choices of edge servers [11–13]. In reality, in addition to the task volume of each mobile device which may be different, there may also be performance differences among MEC servers. Simply considering the relative uniformity of task allocation among servers may cause the performance of high-performance servers to be wasted and aggravate the waiting time and performance wear and tear of less performing servers. In studies involving user task waiting time (stay time), they are mainly divided into two forms: computational time accrual for queueing tasks [14] [15] [13] and the use of the M/M/1 queuing model [16] [17]. For the latter, researchers have ignored the effect of the scheduling algorithm on the Markov property of the M/M/1 queuing model, i.e., subjective task scheduling that undermines the principle of no posteriority of the task arrival process.

Require: C_{max}, $B \longrightarrow \infty$, $\delta = 1$, $\eta = 0$, σ
1: **for** $s_i \in S$ **do**
2: Generate uniformly distributed random variables λ_i, μ_i, ζ_i from [0,1)
3: **if** $\mu_i \leq \lambda_i$ **then**
4: $\mu_i \leftrightarrow \lambda_i$
5: **end if**
6: $C_{i,n} = \delta((a_j/B) + (l_{j,i}/v) + (1/\mu_i(1-\zeta_i))) + \eta \cdot \kappa \cdot \phi_{i,n}$
7: **end for**
8: Generate normally distributed random variables α_i, β_i with (1, 0.5)
9: **while** $t \leq T$ **do**
10: **for** $s_i \in S$ **do**
11: $\theta_i \longleftarrow Beta(\alpha_i, \beta_i)$
12: **end for**
13: $s_j(t) \longleftarrow argmax\, \theta_i \forall s_i \in S$
14: $n_j = n_j + 1$
15: Generate uniformly distributed random variable U from [0,1)
16: $c_{j,t} = -\ln(1-U)/\mu_j(1-\zeta_j)$
17: $r_j{}' = 1 - p = 1 - (c_{j,t}/c_{\max})(p \leq 1)$
18: $r_j \longleftarrow r_j + \sigma(r_j{}' - r_j)$
19: $\mu_j \longleftarrow \mu_j + (r_j{}'/n_j)$
20: $(\alpha_k, \beta_k) = \begin{cases} (\alpha_k, \beta_k) \text{if } s_j(t)=k \\ (\alpha_k + r_j, \beta_k + 1 - r_j) \text{if } s_j(t) = k \end{cases}$
21: $t = t + 1$
22: **end while**

Algorithm 1: Thompson sampling nonstationary (TSNS).

Queuing systems generally consist of customers, service desks, and queuing rules [18]. Under conditions independent of other factors, a customer's arrival satisfies Poisson distribution, the service time satisfies negative exponential distribution, and a single service desk processes the task of the customer, those situations that can be represented by the M/M/1 model [18–20]. A customer's arrival is usually independent of others, while the service desk is responsible for solving the task requests of arriving customers. The processing time of specific tasks is influenced by stochastic factors such as the nature of the customer's task and the service desk. When we use the algorithm to schedule the user assignment process in the edge environment, customer arrivals will no longer follow the Poisson distribution, and continuing to use the M/M/1 model at this point seems to deviate from reality. Regarding the G/M/1 model, the process of customer's arrival is not restricted, which is "general arrival," and the service process still follows a negative exponential distribution, which considers the randomness of customer tasks and service desks [18–20].

In the actual edge user assignment process, we reduce the impact of the scheduling algorithm on the Markov property of the edge server's task arrival process by applying the G/M/1 queuing theory model. And a nonstationary factor is added to the MAB algorithm framework to consider the impact of the edge server's task processing capacity fluctuation on the computation delay. In an attempt to improve load balancing in the edge user assignment scenario to mitigate the DDoS attack process, we further consider factors such as the number of tasks offloaded by edge users, possible performance differences among edge servers, and more in-depth consideration of the specific task processing of each edge node. To enhance the experiment's credibility, we used a real dataset from the Central Business District (CBD) of Melbourne [21] and compared it extensively with existing studies, and the experimental results verified the effectiveness of the algorithm. The main contributions are as follows:

(i) We attempt to improve the load balancing for edge user allocation in edge computing to slow down the DDoS attack process

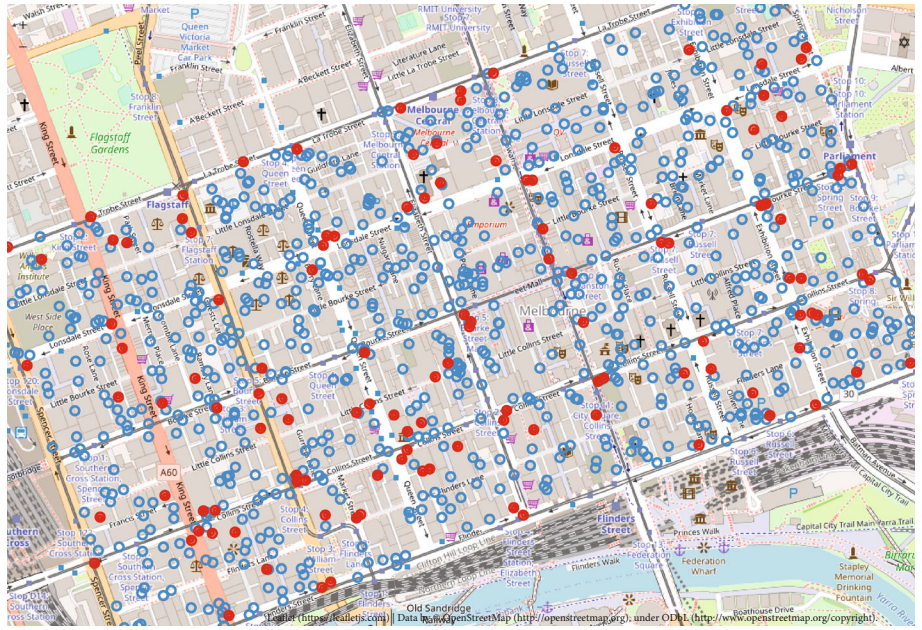
(ii) This is the first attempt to study the EUA problem through the MAB algorithm framework in 5G ultra-dense cellular networks, considering the processing of specific tasks in each edge node. The number of users in edge nodes is no longer considered solely

(iii) This is the first attempt to introduce the G/M/1 queuing theory model to the MEC system, considering the impact of scheduling algorithms on the Markov property of the actual task arrival process. And the performance is experimentally evaluated on a widely used real-world dataset

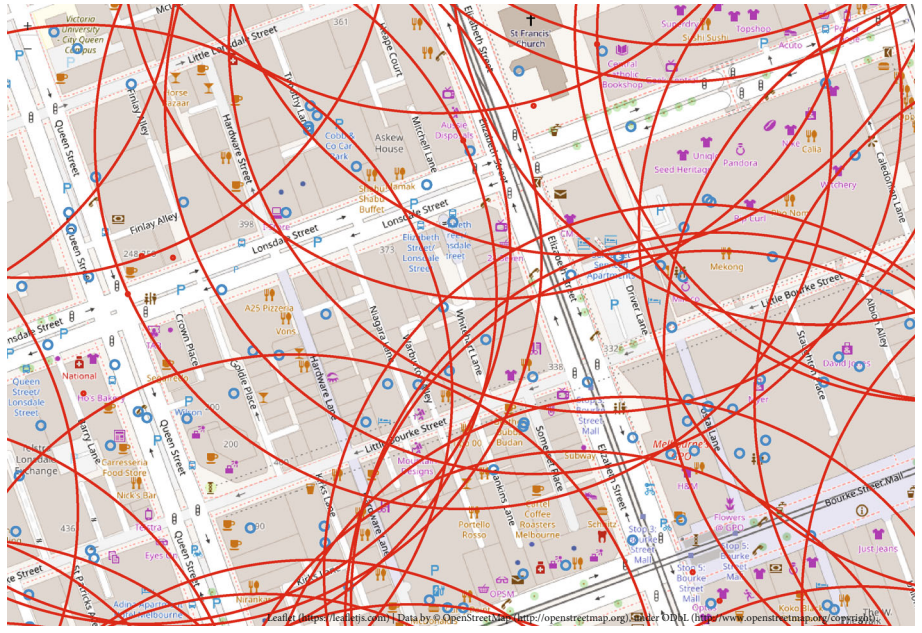
The rest of the paper is organized as follows. Section 2 presents related work. Section 3 offers the system model. Section 4 proposes the Thompson sampling nonstationary (TSNS) algorithm. Section 5 designs experiments and evaluates the algorithm's performance, and Section 6 concludes the paper.

2. Related Work

Currently, relatively little research has been done on DDoS attack in edge computing, mainly in edge collaboration, attack identification, and defense [8, 22–27]. The literature [8] studied the DDoS attack mitigation problem in edge computing, proved its NP-hardness, and proposed a game-theoretic approach to solving the problem. The literature [22] considered mitigating the DDoS attack process by balancing the incoming control plane's total traffic and inducing the attack initiator to stop the attack. The literature [23] designed an adaptive traffic scheduling algorithm to enhance collaboration among edge nodes and thus reduce DDoS attack. The literature [24] developed an intrusion detection and defense method for edge environments by learning the original data distribution through Deep Convolution Neural Network (DCNN) and building defense through Q-network algorithm. In a software-defined network (SDN), [25] established initial detection of intrusion based on entropy and further accurate detection by integrated learning, reducing communication overhead and attack detection latency. From the perspective of smart cities, [26] on the fractional-level fusion of multimodal biometrics effectively improves recognition accuracy and [27]



(a) Point distribution



(b) Relationships

Figure 2: Dataset for Melbourne's Central Business District.

proposed a data encryption technique applicable to the IoT, etc.

The low latency of edge computing is fundamental for users to execute resource-intensive and latency-sensitive applications on edge devices. It is a crucial factor affecting the QoE of user experience [4]. In the study of resource allocation for computational offloading, with the goal of latency optimization, [28–31] schedule computational tasks through a Markov decision process, [14, 32, 33] consider game theory to obtain the best strategy for task offloading, and [11, 12, 34–39] consider algorithms such as reinforcement learning to solve problems related to resource allocation. The literatures [13, 15–17] introduced the MAB algorithm framework to learn online to adjust task allocation in real time. Among them, only the literatures [11–14, 34] consider the load balancing problem of edge servers from the perspective of resource allocation.

The literature [14] proposed a decentralized learning algorithm from a game-theoretic perspective, considering a relatively uniform number of users allocated across edge servers. However, the experimental design with the same upper bound of acceptable cost for users may deviate from reality. From the perspective of on-edge computing, [11] transformed the offloading and load balancing problem into

Table 1: MEC servers.

Parameter	s_1	s_2	s_3	s_4	s_5
μ	0.4250	0.8865	0.6233	0.9485	0.9973
ζ	0.0735	0.7655	0.2477	0.8558	0.9471
Parameter	s_6	s_7	s_8	s_9	s_{10}
μ	0.7121	0.9000	0.8639	0.6185	0.8444
ζ	0.9921	0.0281	0.9425	0.4937	0.3638
Parameter	.....	s_{122}	s_{123}	s_{124}	s_{125}
μ	.....	0.7192	0.9712	0.1817	0.3955
ζ	.....	0.8388	0.2631	0.6614	0.4781

a mixed-integer nonlinear programming problem and changed the problem into two subproblems for optimization. The literature does not seem to consider the effect of the waiting factor, and the possible case of multiple tasks appearing at the same node is not further explored. It is only described from a collision perspective. The literature [34] introduced fiber-wireless (Fi-Wi) technology to enhance the signals of vehicular edge computing networks (VECNs), which in turn use software-defined networking (SDN) to achieve load balancing. The literature considers the possibility of task assignment locally, at the edge nodes or in the cloud. Still, the impact of the coverage of the signaling edge nodes may be neglected in the selection process of the offload servers, and task processing at the edge nodes seems to lack consideration of congestion factors. The literature [39] utilized multipath TCP to increase application throughput and used reinforcement learning-empowered multipath manager to address the buffer congestion problem further. In the literature [12], on the premise of determining the set of optional edge service nodes for each mobile device users (MDUs), the situation of the user devices to be assigned to each edge node and their computational capabilities were considered comprehensively, and new devices were assigned to the edge server with less computational pressure accordingly. The algorithm design process seems to ignore the influence of congestion factors within the edge nodes, and the optimal edge server may deviate from the actual scenario only in terms of transmission and computation delay; i.e., there may be a large waiting delay after the task arrives at the redistributed edge node. In addition, there may be a significant task assignment delay. Uncertainty decision-making is an essential challenge in machine learning, and the MAB algorithm is a common framework for solving this problem, where each MEC server is considered an arm [40]. The literature [13] proposed a utility table-based MAB algorithm with online learning to adjust the workload allocation in real time and update the feedback signal after task allocation through the utility table to determine the optimal solution. The literature mainly considers load balancing from cloud-edge collaboration and gives less consideration to task allocation among edge servers.

As we know, the DDoS attack problem is currently a hot topic of research in network security, and relatively little research has been conducted from the perspective of edge computing. In edge computing, considering the load balancing of edge user allocation, we make the first attempt to study the EUA problem in 5G ultradense cellular networks through the MAB algorithm framework, focusing on the processing of specific tasks at each edge node. We made the first attempt to introduce the G/M/1 queuing theory model into the MEC system, considering the impact of the scheduling algorithm on the Markov property of the actual task arrival process.

3. System Model

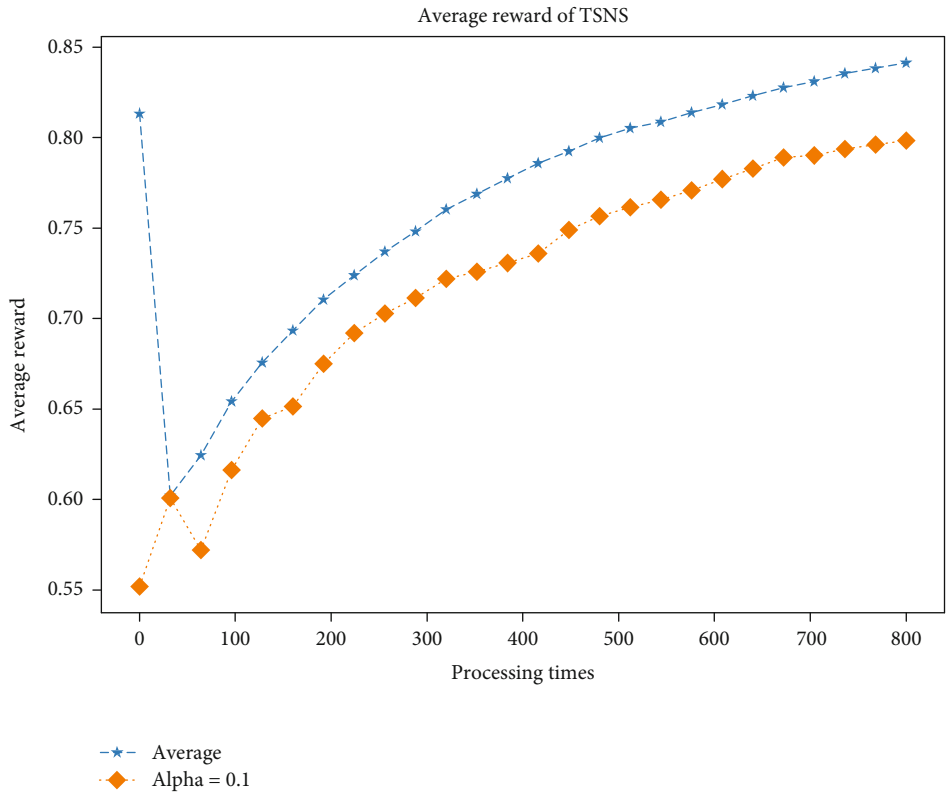
In the ultradense cellular network scenario, in the edge user allocation process, as in Figure 1, we use $s_i \in S$ to denote the set of MEC servers and $u_i \in U$ to represent the set of user devices. In edge computing, in addition to service requests from regular users, DDoS attack can launch frequent task requests to the edge server by controlling multiple IoT devices in the service range. Considering the influence of the scheduling algorithm on the Markov property of the resource allocation process, we assume that the task arrival process follows a general distribution and the service time follows a negative exponential distribution. Since each MEC server has different task arrival and service capacity and there is no restriction on queue length and task origin, the task offloading process can be represented by the G/M/1 queueing theory model. We denote by ζ_i the task arrival impact factor per unit time of server s_i, which can be obtained by solving the scheduling process and by μ_i the average service rate of server s_i. Every time a task assignment is made, the average service rate of the selected server is updated by a nonstationary method.

We consider $C_{i,n}$ to denote the cost of processing task n for server i and $D_{i,n}$ and $E_{i,n}$ to denote the corresponding service latency and energy consumption. Therefore, $C_{i,n}$ is computed as follows:

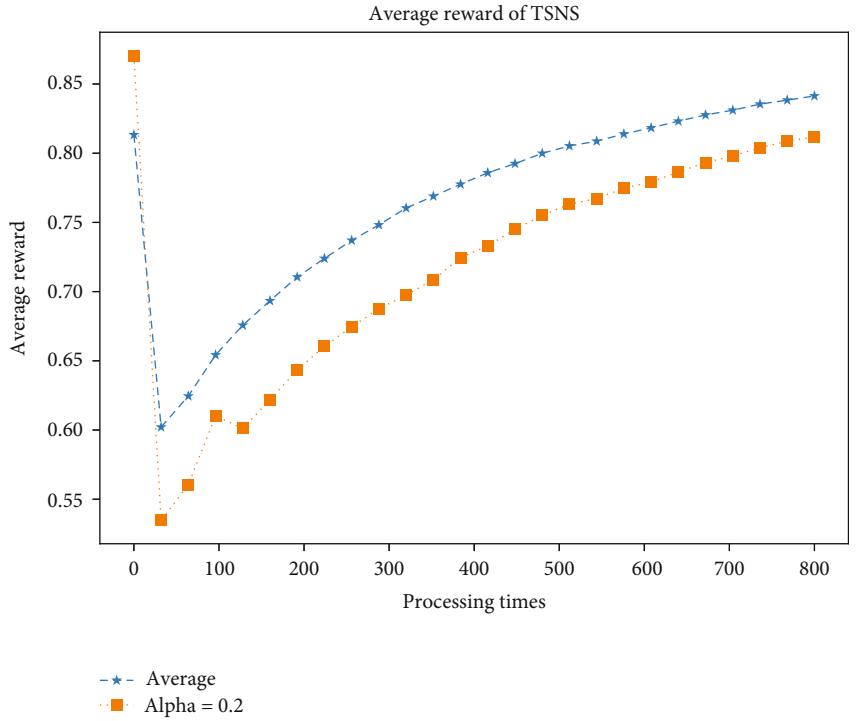
$$C_{i,n} = \delta \cdot D_{i,n} + \eta \cdot E_{i,n}. \tag{1}$$

In the formula, δ and η, respectively, represent the weight of delay and energy consumption in the cost, $\delta + \eta = 1$.

We understand that in 5G ultradense cellular network architecture, the physical distance between microcell BSs is typically between 100 and 200 m [6]. The delay can be further subdivided into transmission delay, propagation delay, waiting delay, and computation delay. The sum of the waiting and computation delays is the delay of the task staying in the system. The signal strength decreases from the central node in all directions [5]. We may assume that the effective signal coverage of each edge node is 200 m (edge servers beyond the signal range can be selected, but the selection cost is relatively high [5]), and on this basis, we consider the limited nature of each user when selecting an edge server. And since the physical distance $l_{i,j}$ of the computing task from the user end u_i to the edge node s_j generally does not exceed 200 m, its actual propagation delay will be at the microsecond level. We usually use the ratio of task volume a_j

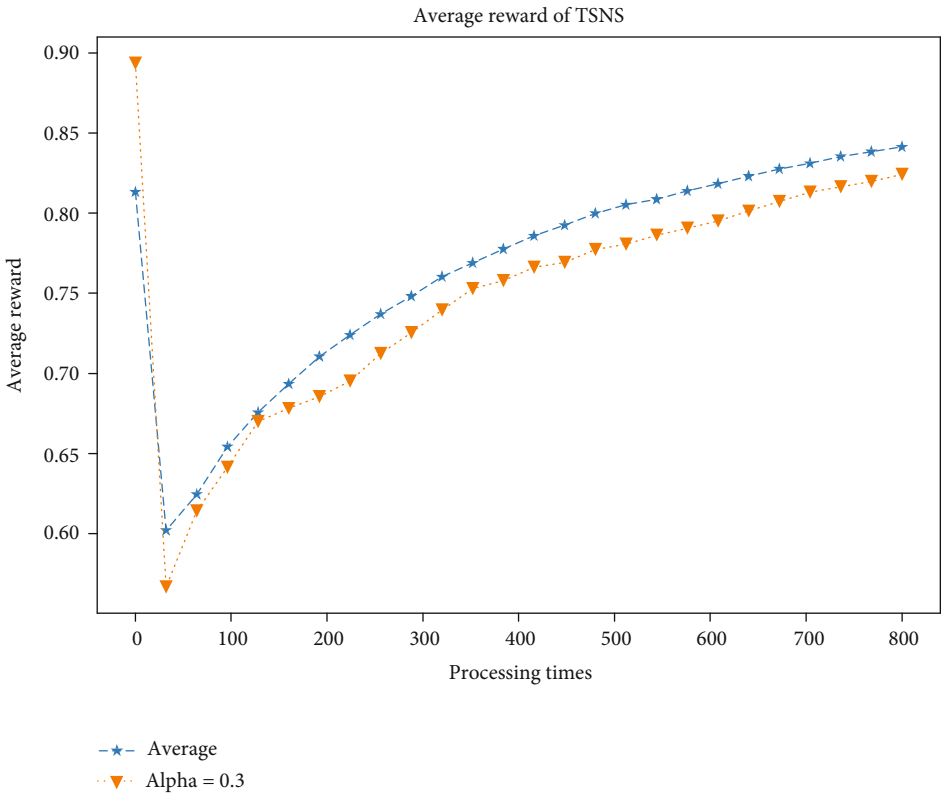


(a) $\sigma = 0.1$

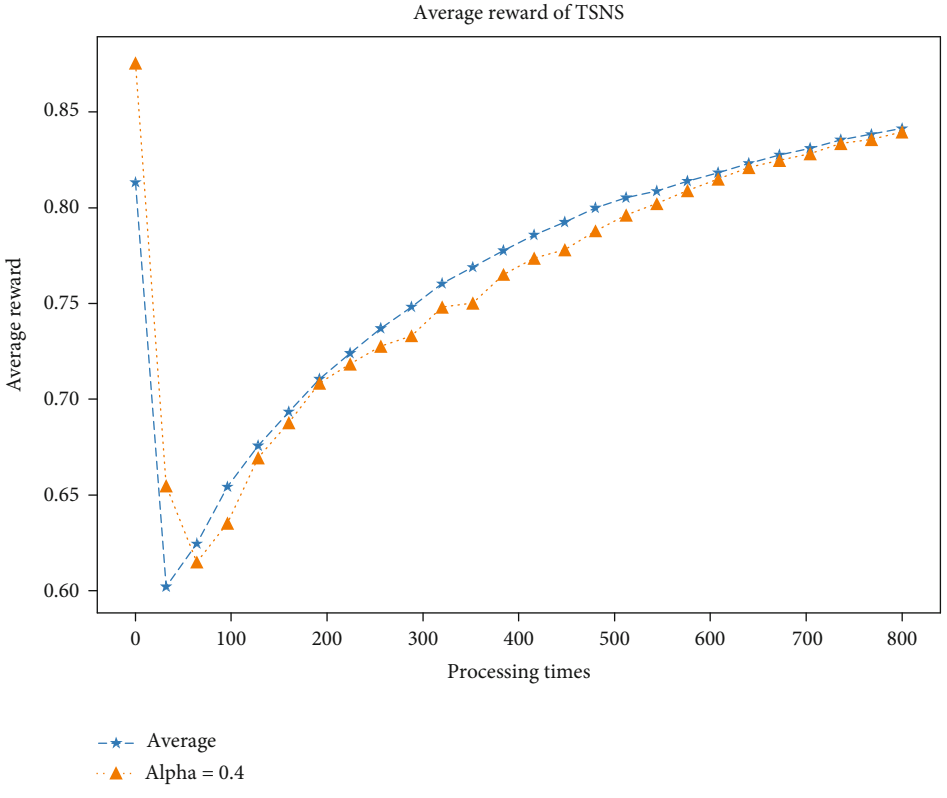


(b) $\sigma = 0.2$

Figure 3: Continued.

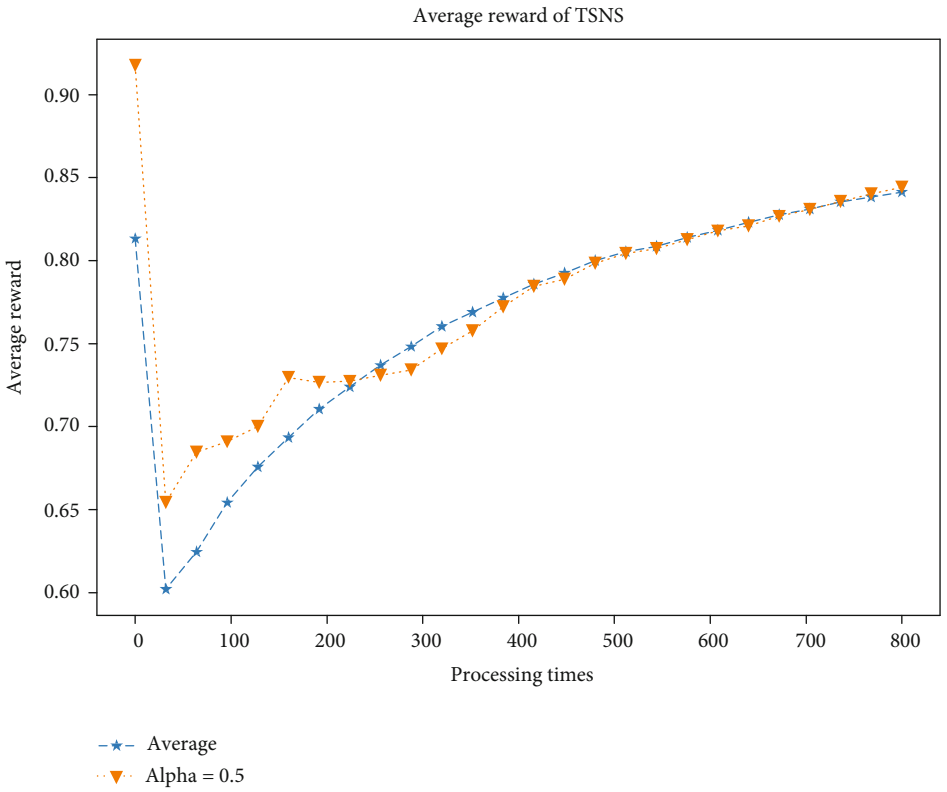


(c) $\sigma = 0.3$

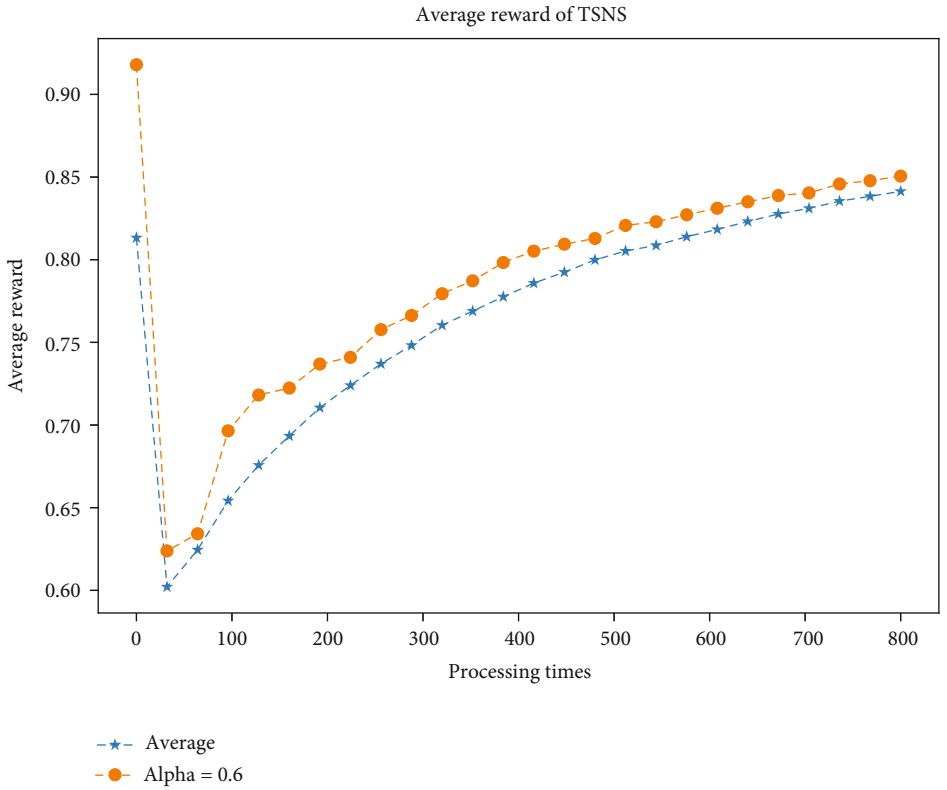


(d) $\sigma = 0.4$

Figure 3: Continued.

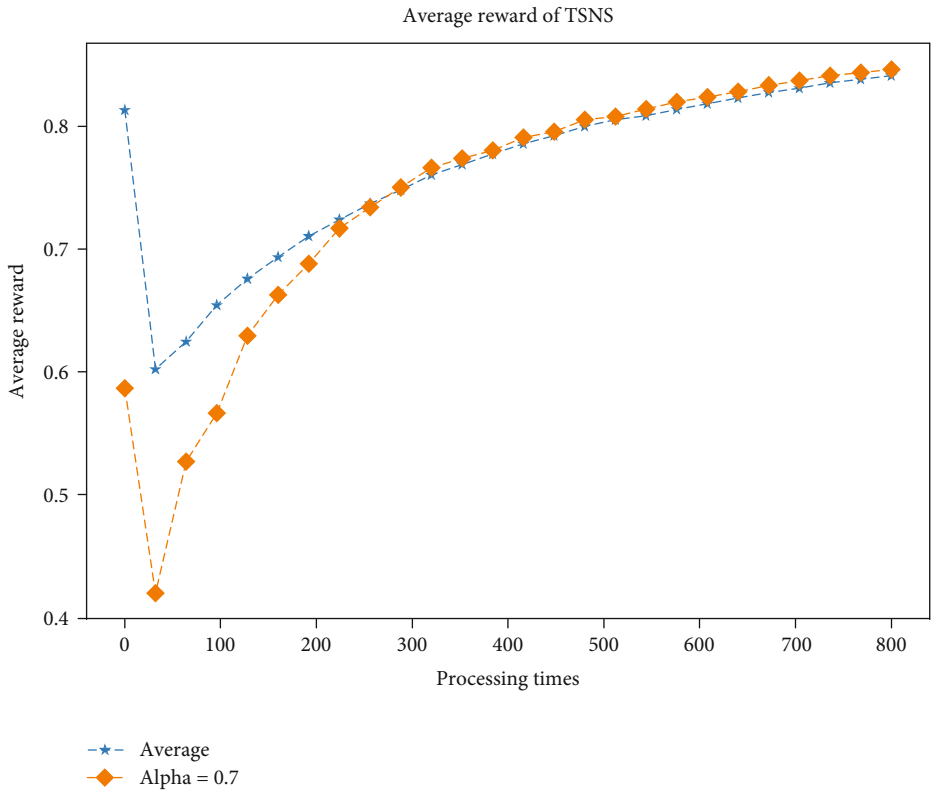


(e) $\sigma = 0.5$

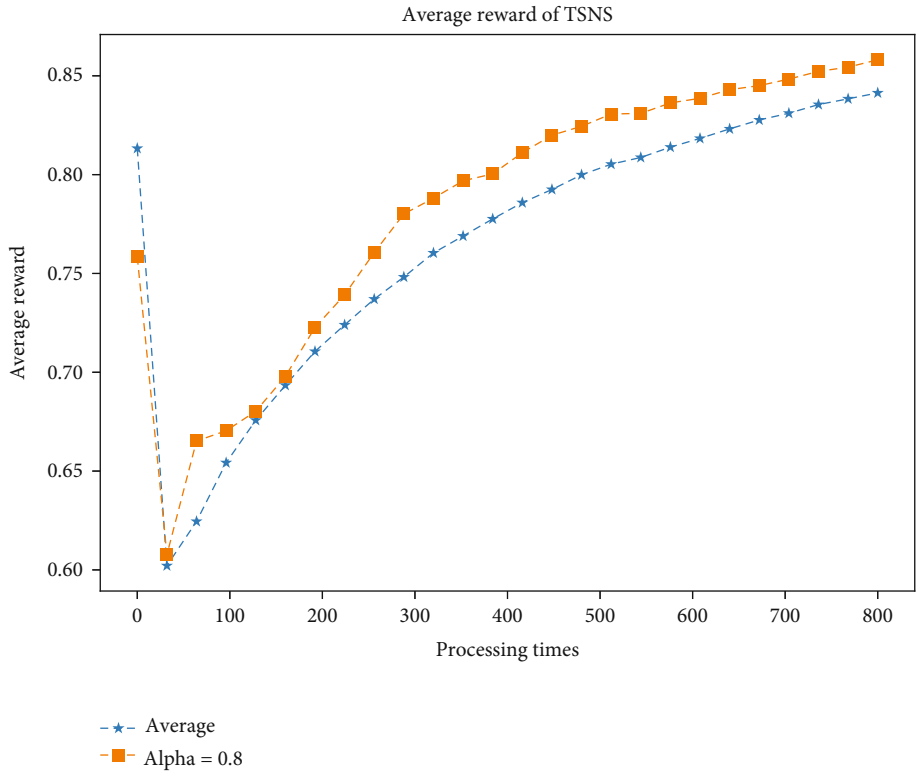


(f) $\sigma = 0.6$

Figure 3: Continued.

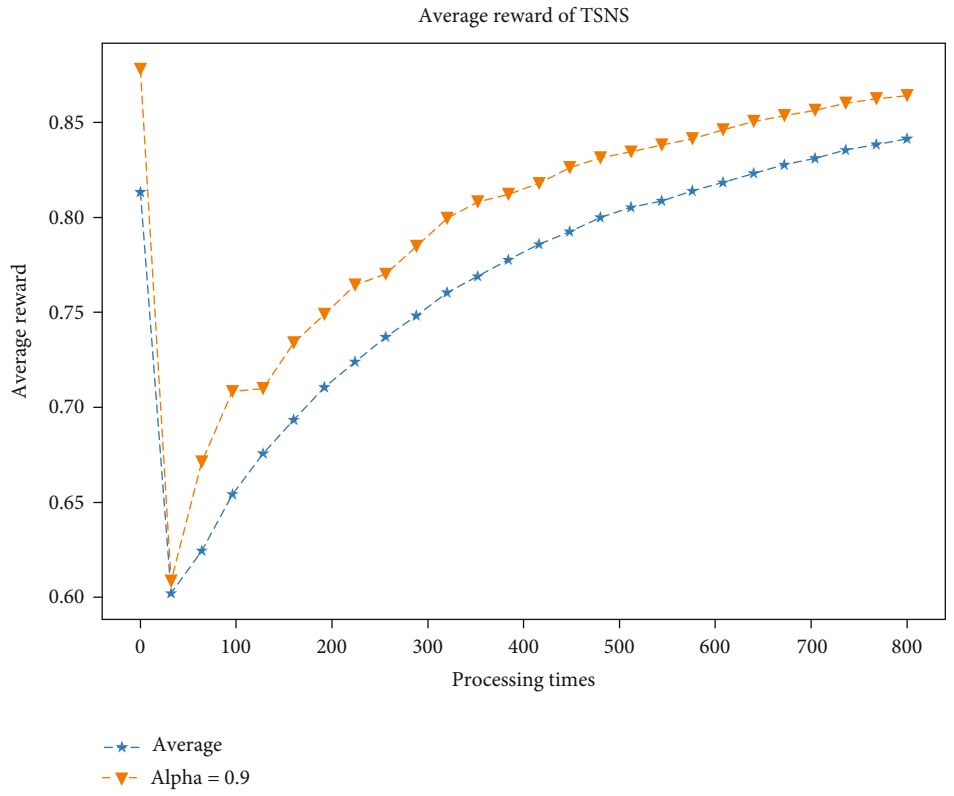


(g) $\sigma = 0.7$



(h) $\sigma = 0.8$

Figure 3: Continued.



(i) $\sigma = 0.9$

Figure 3: Different trends of the average reward in different alpha.

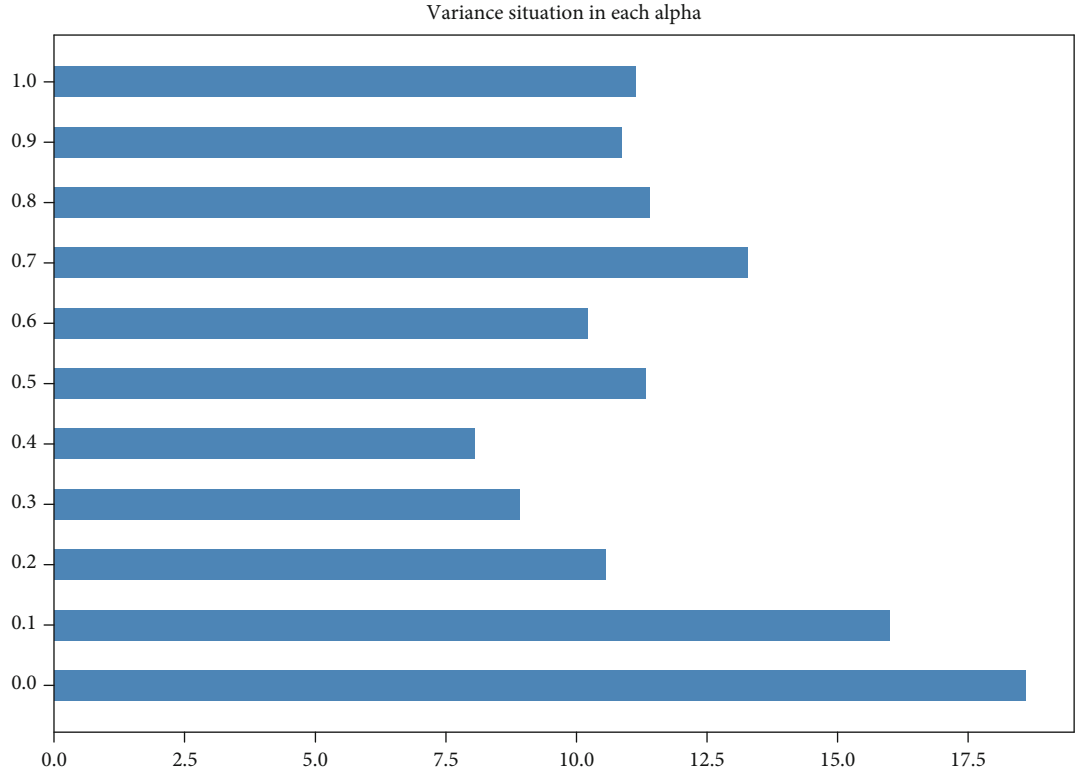


Figure 4: The variance in the TSNS algorithm.

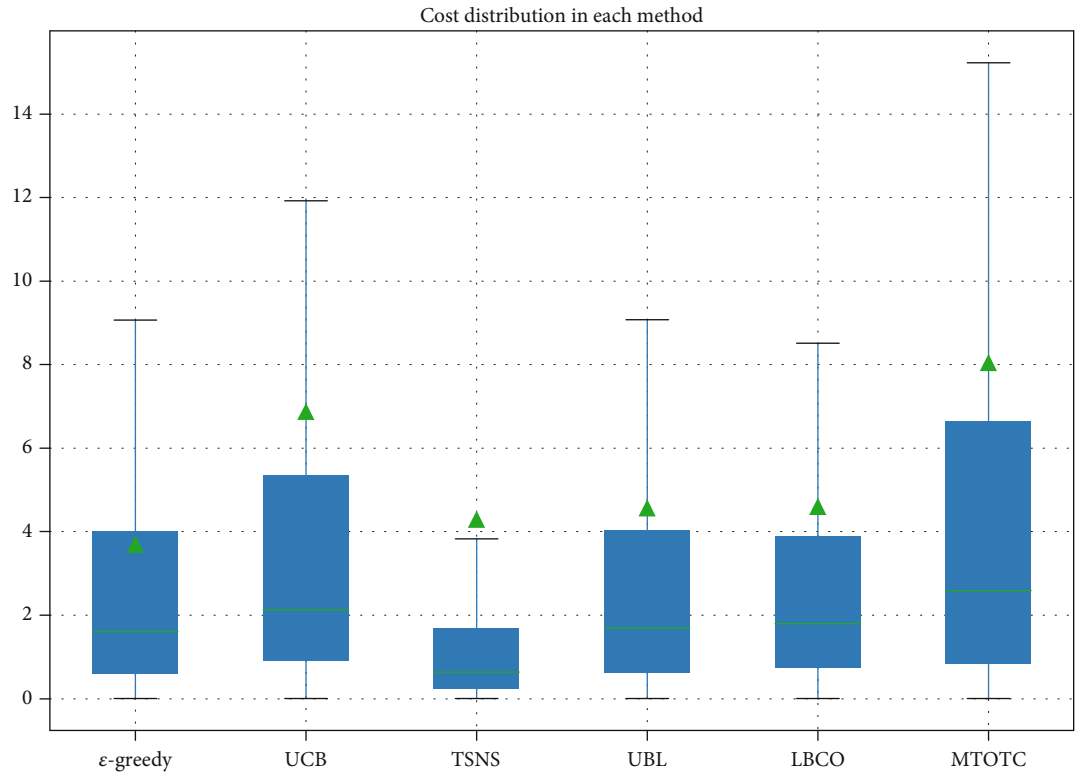


Figure 5: Boxplot for the cost distribution.

to channel bandwidth B to express the transmission delay. In the ultradense cellular network structure, the case of transition node forwarding tasks is basically nonexistent; i.e., the transmission delay of tasks will also be close to a subtle level. According to the queuing theory [18–20], the stay time T obeys the distribution $P\{T \le t\} = 1 - e^{-\mu(1-\zeta)t}$, whose average stay time is $1/(\mu(1-\zeta))$, and the actual stay time of each task can be randomized by the distribution function. We use κ to represent the energy consumption influencing factor that comprehensively considers power, signal-to-noise ratio, and other factors. Let $\phi_{i,n}$ denote the task size of task n in server s_i, and the energy consumption is $E_{i,n} = \kappa \cdot \phi_{i,n}$ [41, 42]. Further, we can get the following formula:

$$C_{i,n} = \delta\left(\frac{a_j}{B} + \frac{l_{j,i}}{v} + \frac{1}{\mu_i(1-\zeta_i)}\right) + \eta \cdot \kappa \cdot \phi_{i,n}, \tag{2}$$

where a_j denotes the number of tasks to be processed by user device u_j, which in general is equal to $\phi_{i,n}$. B is the channel bandwidth, and a_j/B is the transmission delay. $l_{j,i}$ denotes the physical distance between user device u_j and edge node s_i. v indicates the propagation speed of the task in the channel, which is generally equal to or slightly less than the speed of light, and $l_{j,i}/v$ is propagation delay.

To measure the QoE more specifically, we introduced the MAB algorithm framework. An upper bound on the cost is chosen as $C_{\max}$, and we assume that the cost as a percentage of the given threshold is p. The reward after each selection can be calculated as follows:

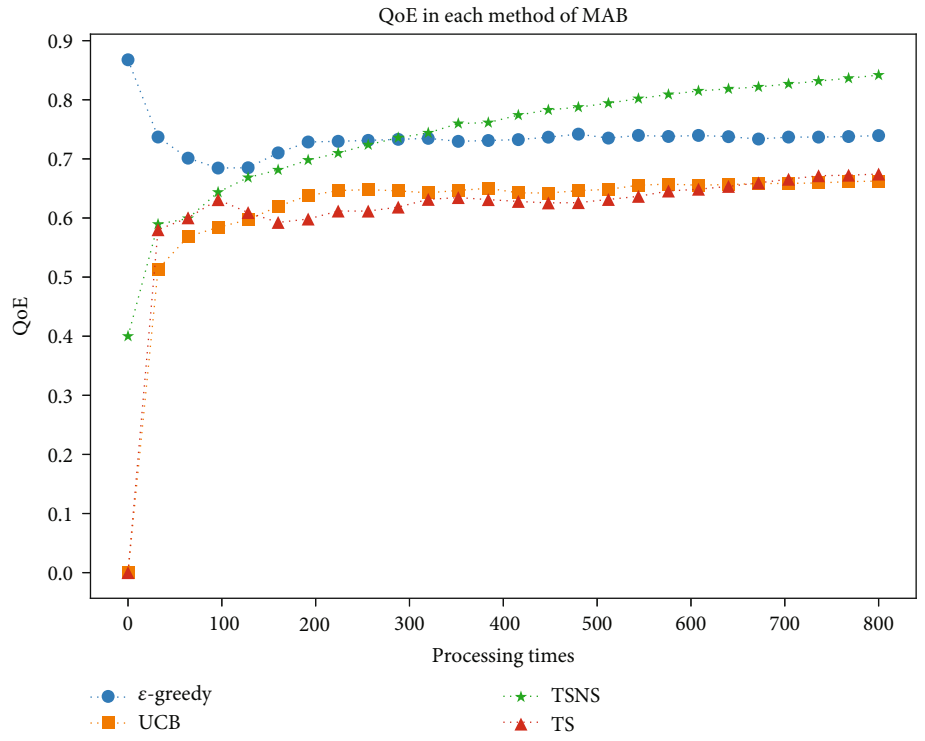
$$R_{i,n} = (1-p) \cdot 1_{(C \le C_{\max})}, \tag{3}$$

where 1 is the indicator function.

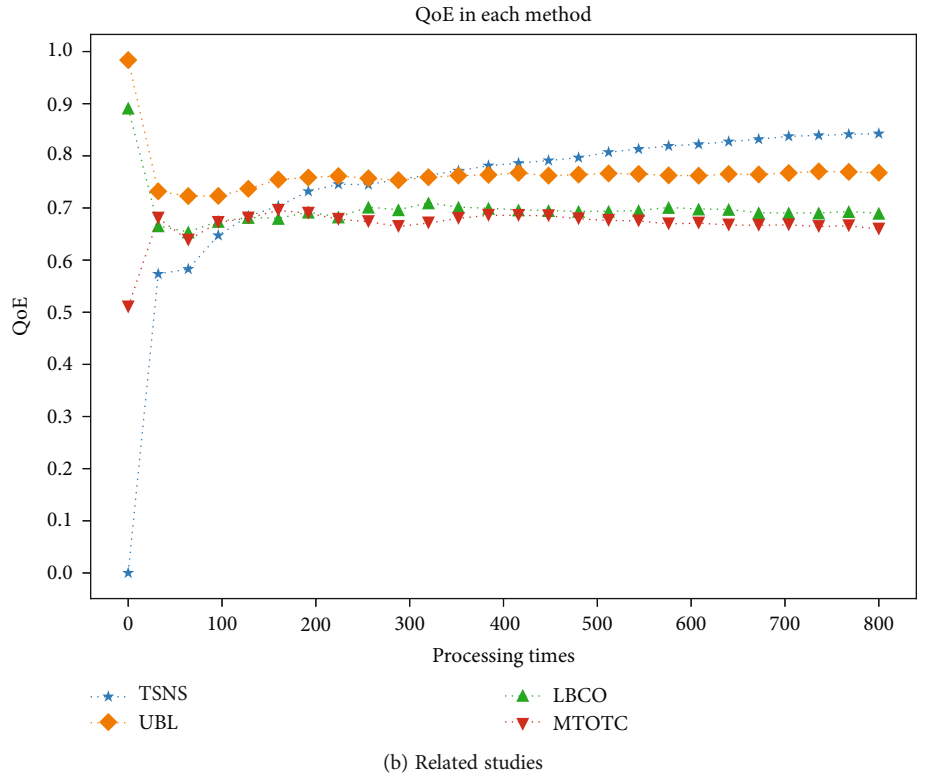
4. Algorithm Design

The MAB model is a simple but compelling algorithmic framework that can make decisions over time in uncertain situations [43]. It simulates an agent, learning new knowledge to optimize selection decisions.

We know that considering load balancing in the edge environment is beneficial to slow down the DDoS attack process [8, 22]. We use the MAB algorithm framework to balance the limited task processing latency and cost and offload the tasks to each MEC server as evenly as possible. Each MEC server can be considered an arm of varying nature, and each selection of the arm can be rewarded and cost accordingly. This property is unknown to the task assignor, so we may call it an implicit property. As the number of selections increases, the resource allocation of edge servers will become more rational, and the number of tasks processed per unit time will improve. In addition, considering the complexity and variability of the actual task arrival and processing, the server's performance may also change with the increasing number of selections, and we introduce a nonstationary factor.



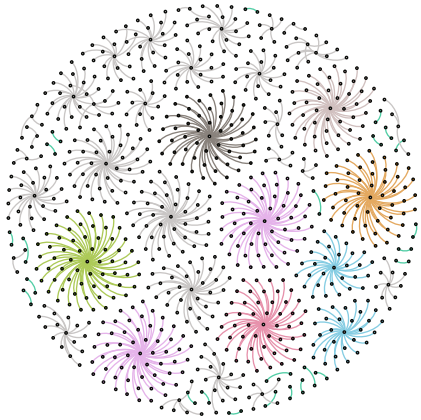
(a) In MAB



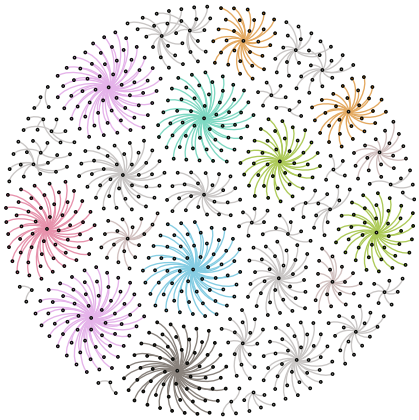
(b) Related studies

Figure 6: QoE values during the selection process.

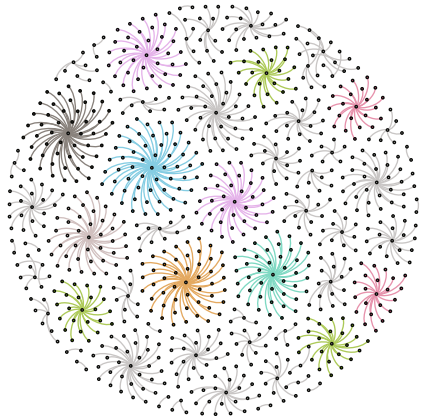
To reduce useless exploration and increase the exploration of the arm with larger pairwise differences, we consider applying the improved Thompson sampling to the MAB algorithm. In the Thompson sampling algorithm, the payoff value of each action follows a beta distribution, with α and β as prior probability parameters. All the arms will generate a random number as payoff value through beta distribution according to their prior probability parameters whenever a selection is made. The system will select the arm with the largest payoff value. The probability distribution law of



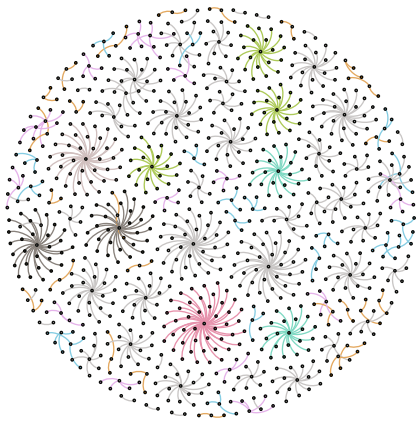
(a) UBL



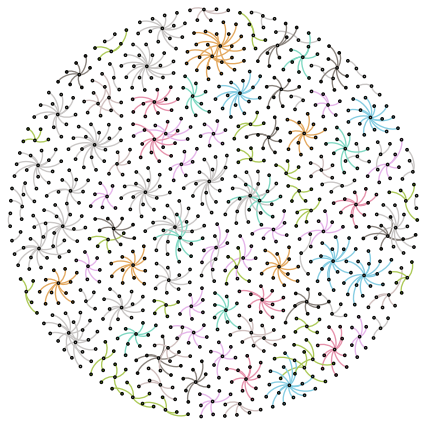
(b) LBCO



(c) Improved ε-greedy

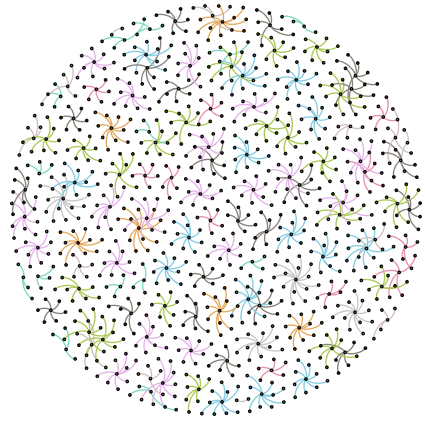


(d) TSNS



(e) MTOTC

Figure 7: Continued.



(f) UCB

Figure 7: Relationship between edge nodes and user devices.

Bernoulli distribution and the probability density of beta distribution are as follows:

$$p(x) = \theta^x (1-\theta)^{1-x}, \quad x = 0, 1, \tag{4}$$

$$p(\theta) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \theta^{\alpha-1}(1-\theta)^{\beta-1}, \quad \theta \in [0, 1], \tag{5}$$

where the two refer to the distribution of returns and the distribution of the parameter θ of the return distribution. $\Gamma(z)$ satisfies the formula

$$\Gamma(z) = \int_0^{\infty} x^{z-1} e^{-x} dx, \quad R(z) > 0. \tag{6}$$

We assume that there are S edge servers, and T tasks are processed in a certain period of time. Each selection will update the distribution. When action k is selected, the return is subject to a Bernoulli distribution with parameter θ_k.The probability of returning 1 is θ_k, and returning 0 is $1-\theta_k, \theta = (\theta_1, \theta_2, \theta_s)$. In round t, select the action $a_t \in \{1, 2, s\}$ will receive a return $r_t \in (0, 1)$. Assuming that θ_k are independent of each other, the prior distribution obeys beta(α_k, β_k), and the posterior distribution obeys beta$(\alpha_k + r_t, \beta_k + 1 - r_t)$.

$$p(\theta_k) \propto \theta_k^{\alpha_k - 1}(1-\theta_k)^{\beta_k - 1}, \tag{7}$$

$$p(\theta_k | r_t) \propto \theta_k^{r_t}(1-\theta_k)^{1-r_t}\theta_k^{\alpha_k-1}(1-\theta_k)^{\beta_k-1} = \theta_k^{\alpha_k+r_t-1}(1-\theta_k)^{\beta_k+1-r_t-1} \tag{8}$$

For each selection made, the parameters of the posterior distribution of the selected arm will be calculated based on its return values. The posterior distribution of the last round can be used as the prior distribution of the next round, and the parameter update rule of the posterior distribution beta is [44]

$$(\alpha_k, \beta_k) = \begin{cases} (\alpha_k, \beta_k) & \text{if } a_t \neq k, \\ (\alpha_k + r_t, \beta_k + 1 - r_t) & \text{if } a_t = k. \end{cases} \tag{9}$$

We use the reward $R_{i,n}$ of the edge nodes after performing the task processing as the QoE measure for the corresponding users. As we analyzed above, the propagation delay and transmission delay under delay segmentation is at the microsecond level, which is negligible compared to the task's computation delay and queuing delay. In contrast, the task processing energy consumption is a weak user experience. To simplify the model, we set $\eta = 0$ and $\delta = 1$, and the channel bandwidth is infinite concerning the task volume and mainly considers the average stay time of the task in the system. After each selection, we add a nonstationary utility learning mechanism [45].

$$Q_{i,n} = Q_{i,n-1} + \gamma(R_{i,n} - Q_{i,n-1}), \tag{10}$$

where γ represents the learning rate in the selection process; i.e., the greater the γ, the greater the importance of the actual reward, and the greater the degree of learning in calculating the utility reward. The updated utility reward is used as the reward value. In particular, after each selection, the average service rate of the selected service desk is optimized to simulate the effect of random factors in the user assignment process.

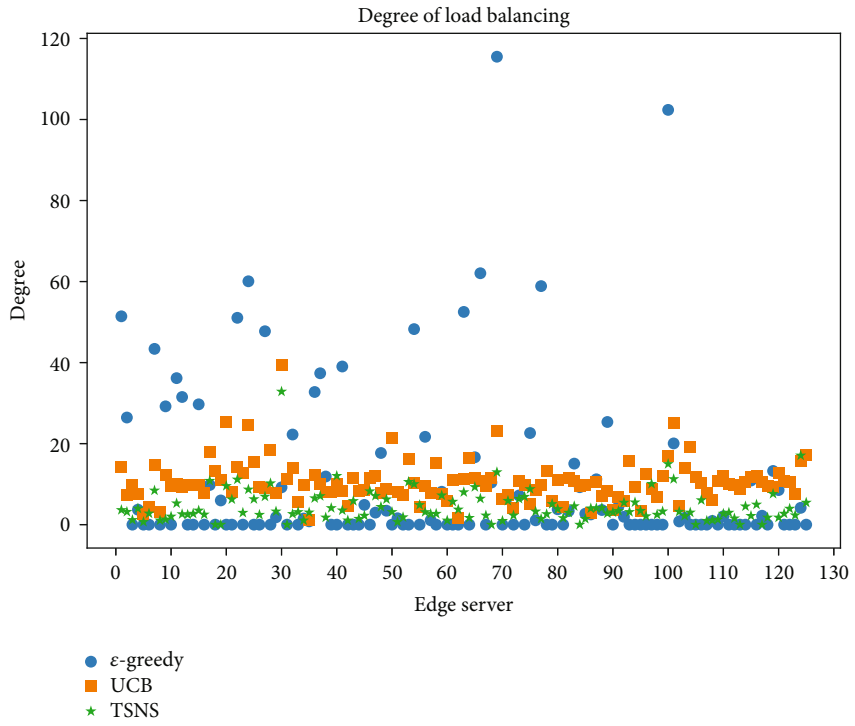
$$\mu_{i,n+1} = \mu_{i,n} + \frac{R_{i,n}}{n}. \tag{11}$$

The arm with the largest parameter θ is considered during each selection, and the calculated reward of the actual choice is used to update the posterior distribution parameters beta. The corresponding regret value is

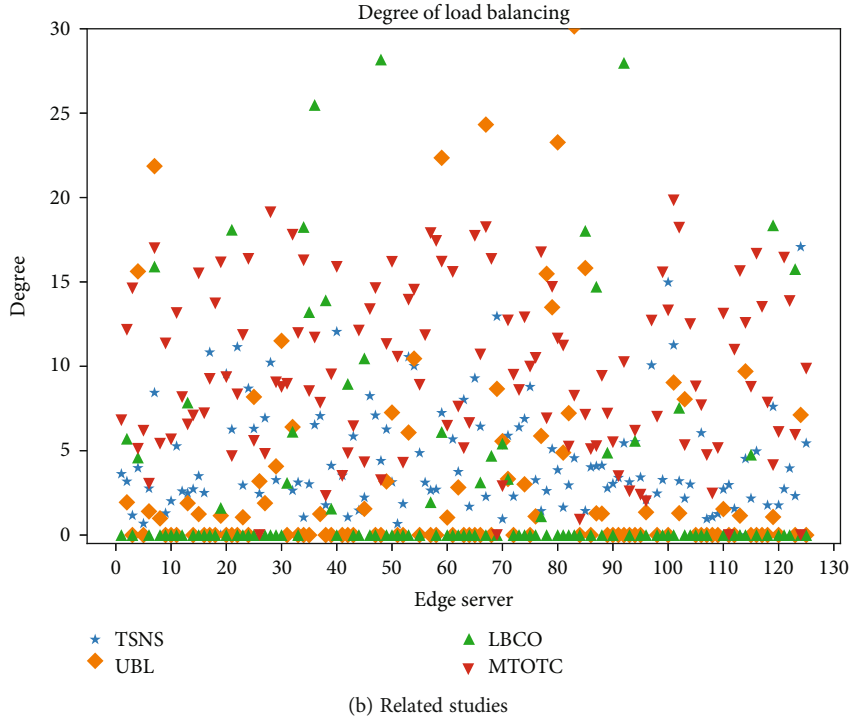
$$R(T) = \sum_{t=1}^{T} C(a_t) - \sum_{t=1}^{T} C_t^*(a_t \in S), \tag{12}$$

where a_t represents the edge server selected for time t and $C(a_t)$ is the corresponding cost of the currently selected server. C_t^* denotes the minimum value of the corresponding cost of each edge server at time t.

The specific idea of the TSNS algorithm is represented in Algorithm 1 in an ordered manner. Each user assignment is



(a) In MAB



(b) Related studies

Figure 8: Accumulated computation time in each edge node.

made that the corresponding service time and stay time are calculated according to the G/M/1 queuing theory model. Service time is an essential statistic for measuring load balancing. Stay time can be used to calculate rewards and, in turn, utility rewards.

We learn and record the specific situation after each task assignment through the MAB algorithm framework, including the actual cost and reward after each user assignment and the actual task processing latency of edge nodes, which can measure QoE and load balancing more precisely and

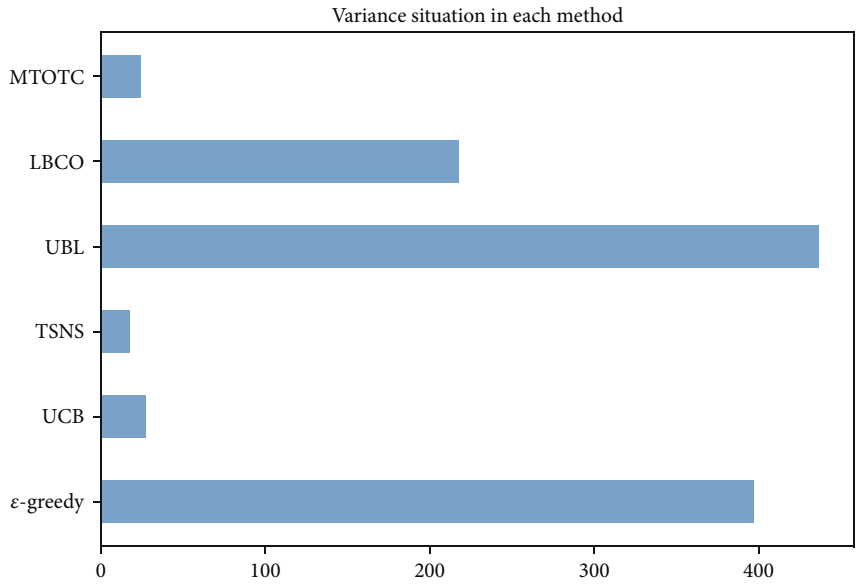


Figure 9: Load profile in each edge node.

effectively. The specific experiments are described in detail in Section 5.

5. Performance Evaluation

In this section, we conducted extensive experiments to evaluate the TSNS algorithm based on a real-world dataset from the CBD of Melbourne (e.g., Figure 2). The excerpted dataset contains 125 service base stations and 816 random users. We model the difference in task volume between users using a normal distribution with mean 10 and variance 2, in which the ratio of the random number to the mean is used as the weight of the effect of task volume on processing latency and use this as the basis for a series of experiments.

5.1. Preferences. Combining the ideas of the Monte Carlo method, we conducted an experimental design. First, we record and calculate the average stay time as the initial property of the corresponding service desk, in which the average service rate and the task arrival impact factor per unit time are solved by a uniform distribution in the interval [0,1), as in Table 1 (the experiment contains but is not limited to the parameters given in Table 1). Based on the nature of queuing theory regarding the G/M/1 model, we obtain the distribution function of the task sojourn time and consider randomizing the essential stay time of the current task at the selected server by the distribution function. For each user's specific task, we assume that it satisfies a normal distribution with a mean of 10 and a variance of 2. The ratio of the random value to the mean is used as the influence of the stay time for the specific task. That is, for each task assignment, the corresponding edge server randomizes the corresponding sojourn time for calculating the reward and, in turn, the utility reward. The calculated utility rewards are used to update the parameters of the posterior distribution beta, which in turn affects the next round of task assignment.

Regarding the initial prior distribution corresponding to each server in the TSNS algorithm, it is considered randomized out through a normal distribution with a mean of 1 and a variance of 0.5. In each task assignment process, the posterior distribution will be used as the prior distribution corresponding to the following selection, and the parameters θ_i will be randomized through the prior distribution. Then, the server with the largest parameters will be selected for the task processing.

First, we assume that the learning parameter $\sigma = 0.5$ and the cost upper bound $C_{\max} = 20$. During the selection process of the simulated edge servers, we obtained the upper quartile (Q_3), median, and lower quartile (Q_1) of the corresponding cost distribution for each method. We calculated the maximum observed value of the upper edge by $Q_3 + 1.5 * (Q_3 - Q_1)$ [46]. Subsequently, we averaged the upper edge observations for all methods and calculated an approximate cost upper bound of 10. Further, we compared the average reward profile under different learning parameters and obtained the average profile after removing the anomalous profile $\sigma = 0$. In multiple experiments, the larger the parameter σ, the better the distribution of the average reward might be, and $\sigma = 0.4$ basically fluctuates up and down around the average curve, as shown in Figure 3. We simulated the user assignment process under different parameters and obtained the variance comparison among the edge nodes, as shown in Figure 4. To balance the load situation of the server, we might as well set it as the experimental parameter. Comparison of cost distribution among methods for the upper cost bound $C_{\max} = 10$ and learning parameter $\sigma = 0.4$ is shown in Figure 5.

5.2. Algorithm Performance. We determine the cost upper bound and learning parameters through the above experiments. Subsequently, we will examine the performance of the TSNS algorithm in terms of user QoE and load balancing by comparing it with classical methods and related work.

(i) Improved ε-greedy: the edge node with the highest utility value is explored or selected with a certain probability. After the algorithm is improved, ε keeps getting smaller, and the exploration probability keeps decreasing as the number of selections increases

(ii) UCB: all optional but not yet selected edge servers are first explored. Subsequently, the edge node with the largest utility value is selected, and the utility value is updated after each selection

(iii) UBL [40]: based on the improvement of the general greedy algorithm, the utility value of the selected edge node is updated after each selection. If the same edge server is selected twice in a row, the utility value of the corresponding server is updated to a temporary value

(iv) LBCO [35]: first, determine the number of mobile devices offloaded to each edge node, consider the different available uplink data rates of the user-side devices and the computing power of the edge nodes, calculate the upload and service times for each task, obtain the set of edge nodes available to the users, calculate the corresponding times, and force each user to select the optimal edge node for the task

(v) MTOTC [27]: each user has partially selectable edge nodes, and the selection probability of all selectable nodes is summed to 1. The stochastic congestion game with incomplete information is performed based on the careful consideration of each user's task type and different task volumes. When the probability of all users selecting an edge node is 1 or the probability within an acceptable error range is greater than the set value, the game stops, and the corresponding edge node is the final choice of users

We evaluated the methodology from two main perspectives.

(i) QoE: the metric is expressed in terms of the average reward earned by users after uninstalling a task and is necessary for measuring service quality

(ii) Load balancing: this metric compares the total number of tasks ultimately served by each edge service but specifically considers the cumulative computation time for task processing in each service. This manuscript's load balancing degree is the primary metric to measure DDoS attack mitigation

In Figure 6, by computing the actual reward after each selection, we obtain a graph of the evolutionary trend of the average reward for each algorithm and, in turn, represent it as the evolutionary trend of the QoE. First, the algorithm was compared with the TS algorithm, which is based on the M/M/1 queuing model and the classical algorithms (improved ε-greedy and UCB) within the framework of the MAB algorithm. We find that the algorithm with the G/M/1 model will significantly outperform the case with the M/M/1 model in terms of QoE performance, having more significant advantages and potential. The algorithm that uses the M/M/1 model is similar to the UCB algorithm but significantly lower than the improved ε-greedy algorithm and the TSNS algorithm. Subsequently, during the comparison with related work, we found that the UBL, LBCO, and MTOTC algorithms reach their QoE peaks relatively quickly and are largely stable. In contrast, the TSNS algorithm suffers from a slow learning ascent. However, as the user assignment process continues, the TSNS algorithm outperforms the other algorithms in terms of QoE overall.

We can find that all algorithms in the MAB algorithm framework fluctuate to some extent at the operation beginning, especially during the first 100 edge user assignments. Because properties, such as the service rate of all servers, are unknown to the algorithm in the MAB framework at the beginning, the quality of user assignment could be gradually improved through continuous selection. Considering the influence of stochastic factors in the actual task arrival and processing process, server performance may also change with time; we introduce a nonstationary factor in the algorithm improvement; i.e., after each task assignment, a reward is calculated based on the task processing process, and the service rate of the edge servers is updated based on the reward.

In experiments, we count the specifics of user selection of edge nodes in different methods and represent them as Figure 7. We can see that large numbers of clusters form the representation graph for each method. The centers of the clusters represent edge servers, while the ends represent users, and the connecting lines between them represent their selection relationships. The size and density of the clusters can reflect the uniformity in selecting edge nodes by users. Among them, UBL, LBCO, and improved ε-greedy algorithms mainly focus on choosing some fixed edge nodes, and fewer edge nodes connect more users. In contrast, the TSNS, MTOTC, and UCB algorithms can distribute edge users more evenly, and the number of users served by each edge node is similar.

However, since the task volume of tasks to be processed by different users and the computational capacity of edge nodes vary, we also need to discuss the task processing of each edge node more specifically.

As in Figure 8, we count the work of each edge server between methods. Where the vertical coordinate represents the accumulated computation time of each edge server, which is expressed as the degree of load, ideally, the degree of load should be essentially similar between edge servers, although there are some fluctuations. This figure shows more intuitively that the load within the UCB, TSNS, and MTOTC algorithms are relatively homogeneous, compared to other algorithms, with slight fluctuations basically around a certain level. To quantify this balance's level more concretely, the changes in stay time are calculated and subsequently expressed as the variance. As shown in Figure 9,

we can conclude that the TSNS algorithm has some advantages in load balancing compared with other algorithms. This advantage is beneficial in resource-limited edge environments, facilitating the mitigation of DDoS attack processes and, in turn, reducing the probability of system breaches.

6. Conclusion

In this paper, to slow down the DDoS attack process in edge computing, we have focused on the EUA problem in a 5G ultradense cellular network scenario and considered improving the load balancing of edge servers while guaranteeing the QoE. To quantify the QoE, we have introduced the MAB algorithm framework and added nonstationary factors to the learning mechanism. Considering the effect of scheduling algorithms on the Markov property of the task arrival process, we have introduced the G/M/1 queueing theory model to EUA for the first time. We have focused on processing specific tasks in each edge server and conducted a series of experiments on real-world datasets, which verified the strength and potential of the algorithm in the target scenario.

In future research, in the context of non-orthogonal multiple access (NOMA) for 5G networks, we will consider more general cases of load balancing of edge demand response under the impact of latency and energy consumption. And we will slow down the process of DDoS attack in edge computing by pursuing load balancing of edge servers. First, we will specifically consider the number and performance of physical machines installed in each edge server and further consider the specific processing process after tasks reach edge servers; subsequently, we will combine cloud-edge collaboration and collaboration among edge nodes to set the threshold to determine whether users need to receive cloud services; more importantly, we will considering the performance of the algorithm in three aspects: mobile users, edge infrastructure providers, and edge service providers, considering the QoE, system energy consumption, and DDoS attack mitigation, to make the model more generalized, etc.

Data Availability

The data used to support the findings of this study is cited in the article and can be viewed via the link https://github.com/swinedge/eua-dataset.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: the communication perspective," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, pp. 2322–2358, 2017.

[2] N. Abbas, Y. Zhang, A. Taherkordi, and T. Skeie, "Mobile edge computing: a survey," *IEEE Internet of Things Journal*, vol. 5, no. 1, pp. 450–465, 2018.

[3] P. Mach and Z. Becvar, "Mobile edge computing: a survey on architecture and computation offloading," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1628–1656, 2017.

[4] W. Z. Khan, E. Ahmed, S. Hakak, I. Yaqoob, and A. Ahmed, "Edge computing: a survey," *Future Generation Computer Systems*, vol. 97, pp. 219–235, 2019.

[5] Z. Xu, G. Zou, X. Xia et al., "Distance-aware edge user allocation with QoE optimization," in *IEEE International Conference on Web Services (ICWS)*, pp. 66–74, Beijing, China, Oct 2020.

[6] X. Ge, S. Tu, G. Mao, C. Wang, and T. Han, "5G ultra-dense cellular networks," *IEEE Wireless Communications*, vol. 23, no. 1, pp. 72–79, 2016.

[7] R. Roman, J. Lopez, and M. Mambo, "Mobile edge computing, fog et al.:a survey and analysis of security threats and challenges," *Future Generation Computer Systems*, vol. 78, pp. 680–698, 2018.

[8] Q. He, C. Wang, G. Cui et al., "A game-theoretical approach for mitigatingedge DDoS attack," *IEEE Transactions on Dependable and Secure Computing*, 2021.

[9] O. Osanaiye, K. K. R. Choo, and M. Dlodlo, "Distributed denial of service (DDoS) resilience in cloud: review and conceptual cloud DDoS mitigation framework," *Journal of Network and Computer Applications*, vol. 67, pp. 147–165, 2016.

[10] G. Cui, Q. He, X. Xia et al., "Demand response in NOMA-based mobile edge computing: a two-phase game-theoretical approach," *IEEE Transactions on Mobile Computing*, 2021.

[11] Y. Dai, D. Xu, S. Maharjan, and Y. Zhang, "Joint load balancing and offloading in vehicular edge computing and networks," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4377–4387, 2019.

[12] W. Z. Zhang, I. A. Elgendy, M. Hammad et al., "Secure and optimized load balancing for multitier IoT and edge-cloud computing systems," *IEEE Internet of Things Journal*, vol. 8, no. 10, pp. 8119–8132, 2021.

[13] P. Dai, Z. Hang, K. Liu et al., "Multi-armed bandit learning for computation-intensive services in MEC-empowered vehicular networks," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 7, pp. 7821–7834, 2020.

[14] F. Zhang and M. M. Wang, "Stochastic congestion game for load balancing in mobile-edge computing," *IEEE Internet of Things Journal*, vol. 8, no. 2, pp. 778–790, 2021.

[15] P. Zhao, H. Tian, K. Chen, S. Fan, and G. Nie, "Context-aware TDD configuration and resource allocation for mobile edge computing," *IEEE Transactions on Communications*, vol. 68, no. 2, pp. 1118–1131, 2020.

[16] S. Misra, S. P. Rachuri, P. K. Deb, and A. Mukherjee, "Multi-armed-bandit-based decentralized computation offloading in fog-enabled IoT," *IEEE Internet of Things Journal*, vol. 8, no. 12, pp. 10010–10017, 2021.

[17] S. Ghoorchian and S. Maghsudi, "Multi-armed bandit for energy-efficient and delay-sensitive edge computing in dynamic networks with uncertainty," *IEEE Transactions on Cognitive Communications and Networking*, vol. 7, no. 1, pp. 279–293, 2021.

[18] Y. Hu, *Operational Research Course*, Tsinghua University Press, Beijing, China, 1998.

[19] U. N. Bhat, "An introduction to queueing theory: modeling and analysis in applications," *Boston, MA: Birkhäuser*, vol. 36, 2008.

[20] J. F. Shortle, J. M. Thompson, D. Gross, and C. M. Harris, *Fundamentals of Queueing Theory*, John Wiley & Sons, Hoboken, New Jersey, U.S, 2018.

[21] P. Lai, Q. He, M. Abdelrazek et al., "Optimal edge user allocation in edge computing with variable sized vector bin packing," in *16th International Conference on Service-Oriented Computing (ICSOC2018)*, pp. 230–245, Hangzhou, China, Nov 2018.

[22] R. S. Silva, C. C. Meixner, R. S. Guimarães et al., "REPEL: a strategic approach for defending 5G control plane from DDoS signalling attacks," *IEEE Transactions on Network and Service Management*, vol. 18, no. 3, pp. 3231–3243, 2021.

[23] Y. Li, Y. Zhao, J. Li, X. Yu, Y. Zhao, and J. Zhang, "DDoS attack mitigation based on traffic scheduling in edge computing-enabled TWDM-PON," *IEEE Access*, vol. 9, pp. 166566–166578, 2021.

[24] Z. Liu, X. Yin, and Y. Hu, "CPSS LR-DDoS detection and defense in edge computing utilizing DCNN Q-learning," *IEEE Access*, vol. 8, pp. 42120–42130, 2020.

[25] S. Yu, J. Zhang, J. Liu, X. Zhang, Y. Li, and T. Xu, "A cooperative DDoS attack detection scheme based on entropy and ensemble learning in SDN," *EURASIP Journal on Wireless Communications and Networking*, vol. 2021, 21 pages, 2021.

[26] V. Rajasekar, B. Predić, M. Saracevic et al., "Enhanced multimodal biometric recognition approach for smart cities based on an optimized fuzzy genetic algorithm," *Scientific Reports*, vol. 12, pp. 1–11, 2022.

[27] M. H. Saračević, S. Z. Adamović, V. A. Miškovic et al., "Data encryption for Internet of Things applications based on Catalan objects and two combinatorial structures," *IEEE Transactions on Reliability*, vol. 70, no. 2, pp. 819–830, 2021.

[28] V. Di Valerio and F. Lo Presti, "Optimal virtual machines allocation in mobile femto-cloud computing: an MDP approach," in *IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, pp. 7–11, Istanbul, Turkey, April 2014.

[29] M. G. R. Alam, M. M. Hassan, M. Z. Uddin, A. Almogren, and G. Fortino, "Autonomic computation offloading in mobile edge for IoT applications," *Future Generation Computer Systems*, vol. 90, pp. 149–157, 2019.

[30] X. Qiu, L. Liu, W. Chen, Z. Hong, and Z. Zheng, "Online deep reinforcement learning for computation offloading in blockchain-empowered mobile edge computing," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 8, pp. 8050–8062, 2019.

[31] J. Liu, Y. Mao, J. Zhang, and K. B. Letaief, "Delay-optimal computation task scheduling for mobile-edge computing systems," in *IEEE International Symposium on Information Theory (ISIT)*, pp. 1451–1455, Barcelona, Spain, July 2016.

[32] K. Zhang, Y. Mao, S. Leng, S. Maharjan, and Y. Zhang, "Optimal delay constrained offloading for vehicular edge computing networks," in *IEEE International Conference on Communications (ICC)*, pp. 1–6, Paris, France, May 2017.

[33] S. M. Shahrear Tanzil, O. N. Gharehshiran, and V. Krishnamurthy, "Femto-cloud formation: a coalitional game-theoretic approach," in *IEEE Global Communications Conference (GLOBECOM)*, pp. 1–6, San Diego, CA, USA, Dec 2015.

[34] J. Zhang, H. Guo, J. Liu, and Y. Zhang, "Task offloading in vehicular edge computing networks: a load-balancing solution," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 2, pp. 2092–2104, 2020.

[35] C. Wang, F. R. Yu, C. Liang, Q. Chen, and L. Tang, "Joint computation offloading and interference management in wireless cellular networks with mobile edge computing," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 8, pp. 7432–7445, 2017.

[36] X. Wang, Y. Han, C. Wang, Q. Zhao, X. Chen, and M. Chen, "In-Edge AI: intelligentizing mobile edge computing, caching and communication by federated learning," *IEEE Network*, vol. 33, no. 5, pp. 156–165, 2019.

[37] T. Wang, Y. Liang, Y. Zhang et al., "An intelligent dynamic offloading from cloud to edge for smart IoT systems with big data," *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 4, pp. 2598–2607, 2020.

[38] T. X. Tran and D. Pompili, "Joint task offloading and resource allocation for multi-server mobile-edge computing networks," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 1, pp. 856–868, 2019.

[39] Y. Cao, R. Ji, L. Ji, G. Lei, H. Wang, and X. Shao, "MPTCP: a learning-driven latency-aware multipath transport scheme for industrial internet applications," *IEEE Transactions on Industrial Informatics*, 2022.

[40] T. Lattimore, *Bandit Algorithms*, Cambridge University Press, Cambridge, United Kingdom, 2020.

[41] M. Kamoun, W. Labidi, and M. Sarkiss, "Joint resource allocation and offloading strategies in cloud enabled cellular networks," in *IEEE International Conference on Communications (ICC)*, pp. 5529–5534, London, UK, June 2015.

[42] W. Labidi, M. Sarkiss, and M. Kamoun, "Energy-optimal resource scheduling and computation offloading in small cell networks," in *International Conference on Telecommunications (ICT)*, pp. 313–318, Sydney, NSW, Australia, April 2015.

[43] A. Slivkins, "Introduction to multi-armed bandits," 2019, http://arxiv.org/abs/1904.07272.

[44] D. J. Russo, B. Van Roy, A. Kazerouni, I. Osband, and Z. Wen, "A tutorial on Thompson sampling," *Foundations and Trends in Machine Learning*, vol. 11, no. 1, pp. 1–96, 2017.

[45] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, MIT press, Cambridge, Massachusetts, United States, 2018.

[46] M. Frigge, D. C. Hoaglin, and B. Iglewicz, "Some implementations of the boxplot," *The American Statistician*, vol. 43, no. 1, pp. 50–54, 1989.

Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 8533687, 12 pages
https://doi.org/10.1155/2022/8533687

Research Article

A Projection-Free Adaptive Momentum Optimization Algorithm for Mobile Multimedia Computing

Lin Wang,[1] **Yangfan Zhou**,[1,2] **Xin Wang**,[3] **Zhihang Ji**,[1] **and Xin Liu**[2]

[1]*School of Information Engineering, Henan University of Science and Technology, Luoyang 471023, China*
[2]*Suzhou Institute of Nano-Tech and Nano-Bionics (SINANO), Chinese Academy of Sciences, Suzhou 215123, China*
[3]*School of Business and Management, Shanghai International Studies University, Shanghai 200083, China*

Correspondence should be addressed to Xin Wang; wangxin@shisu.edu.cn

Received 28 January 2022; Accepted 29 March 2022; Published 20 April 2022

Academic Editor: Xun Shao

In mobile multimedia applications, deep learning has received significant interest. Due to the limited computation and storage resources of mobile devices, however, general training methods are hardly suited for mobile multimedia computing. For this reason, we propose an adaptive momentum training (FWAdaBound) algorithm to reduce computation and storage cost, where the Frank-Wolfe method is employed. Furthermore, we rigorously prove the regret bound in order that $O(T^{3/4})$ can be achieved, where T is a time horizon. Finally, the convergence, cost-reduction, and generalization ability of FWAdaBound are validated through various experiments on public datasets.

1. Introduction

In mobile multimedia applications [1, 2], deep learning is a significant method for multimedia computing [3]. Meanwhile, various deep learning models have been successfully implemented in many important fields, such as convolutional neural network [4, 5], recurrent neural network [6, 7], deep belief networks [8, 9], and industrial internet applications [10]. In order to implement deep learning models in mobile multimedia, the training of deep neural networks is a crucial technology. Moreover, because the computation and storage resources of each mobile device is limited, general training methods of deep neural networks are hardly adapted to mobile multimedia computing. For this reason, how to train rapidly deep learning models with lower computational cost is one of challenging tasks in mobile multimedia applications.

In fact, the training process of deep neural networks can be regarded as an optimization process. For this reason, the design of optimization algorithms is necessary in the training process. Currently, stochastic gradient descent (SGD) is a dominative algorithm for training deep networks. SGD is applied widely over the past years since its good generalization ability and could be implemented easily. Despite SGD having performed well in some applications, however, it converges slowly. To accelerate the convergence of SGD, many researchers have proposed various adaptive momentum algorithms based on gradient descent. Generally, optimizing step size and gradient direction of SGD are two main directions that have been studied by researchers.

SGD often oscillates around the optimal solution when step sizes are fixed. To address this issue, some novel algorithms with adaptive step size have been proposed. AdaGrad [11], RMSProp [12], and Adadelta [13] make step sizes changed adaptively as training process goes on. Besides, the current gradient direction in each iteration is randomly selected, thereby it cannot find the direction to reach the optimal solution in the shortest time. For this reason, historical gradient information has been used to adjust the current gradient direction in many novel algorithms. Moreover, these algorithms often use the first-order momentum to maintain historical gradient information and the second-order momentum to adaptive step size at the same time (Adam [14], AMSGrad [15], and AdaBound [16]). It is gen-

erally believed that algorithms combining first-order and second-order momentum originated from Adam and are Adam-type algorithms.

Being an Adam-type algorithms, AdaBound not only inherits good generalization ability of SGD but also maintains fast convergence rate of Adam. However, like other Adam-type algorithm else, AdaBound also uses higher-order projection operators to handle the case where the iteration point is not within the feasible region. Since the projection operation includes the second-order Euclidean distance calculation or higher-order methods measurement, it has a large computational cost in each iteration. Therefore, algorithms with projection operations like AdaBound become prohibitive when dealing with large-scale problems including massive high-dimensional data. To tackle this problem, we focus on proposing a projection-free algorithm based on AdaBound. In the field of optimization, the Frank-Wolfe method is one of the projection-free technologies, which is commonly used in replacing high-order projection operators with linear searches. Therefore, in this paper, we redesign AdaBound algorithm, which is called FWAdaBound by using the Frank-Wolfe method to reduce computation cost of AdaBound. Moreover, we prove that the FWAdaBound algorithm converges under convex conditions and attain a guaranteed regret bound related to the sublinear correlation of time horizon. In addition, FWAdaBound successfully retains AdaBound's performance on convergence and generalization ability.

In this paper, the summary of our contributions is presented as follows:

(i) We propose the FWAdaBound algorithm based on the Frank-Wolfe method and AdaBound optimization algorithm to eliminate costly projection steps in large-scale problems

(ii) We prove the convergence of FWAdaBound under the online learning framework. Moreover, we also show that the regret of FWAdaBound is $O(T^{3/4})$, where T is a time horizon

(iii) We present various of experiments to validate computation cost reduction of FWAdaBound and show good generalization ability of FWAdaBound on public dataset

The rest of this paper is organized as follows: in Section 2, we review some important related work of FWAdaBound. In Section 3, we introduce preliminary knowledge about optimization object and online learning. In Sections 4, we present some frequently used assumptions and detailed design of FWAdaBound. In Section 5, we prove the convergence of FWAdaBound in theory and obtain the regret bound. In Section 6, we conduct various experiments in detail on public datasets. Finally, we present the conclusion of this paper in Section 7.

2. Related Work

SGD performs linear iteration of decision variables based on gradient. Therefore, SGD is one of the simplest and easiest implemented algorithms in deep learning. It has good generalization ability if labeled training samples are sufficient. However, the slow convergence rate of SGD always makes it difficult to converge to optima under limited labeled training samples. To speed up convergence rate of SGD, the first-order momentum and the second-order momentum based on the gradient are used in optimization algorithms. More specifically, the fist-order momentum of the gradient is used to retain historical information of gradient, and the second-order momentum of gradient is used to make the step size adaptive. The first algorithm combining these two momentums of gradient is Adam, which obtains a faster convergence rate than SGD [14]. However, Reddi et al. found that the convergence proof of Adam was problematic and proposed an improved variant of Adam, called AMSGrad [15]. Moreover, [17] advocated that it is beneficial to consider more past gradients when designing adaptive learning rates, and thereby, they proposed NosAdam.

Despite Adam, AMSGrad, and NosAdam both improving the convergence rate, however, these three algorithms all have lower generalization ability than SGD under sufficient training samples. For this reason, [18] proposed SWATS to improve generalization performance by switching from Adam to SGD in the later stages of training. Although SWATS improves generalization ability for adaptive momentum algorithms, its switching time is difficult to be accurately controlled. Based on works mentioned above, [16] analyzed that unstable and extreme learning rates may lead to the lack of generalization performance of adaptive methods. Moreover, [16] used a dynamic boundary of the learning rate, where the upper and lower limits can smoothly converge to a constant final step size, respectively. Furthermore, the algorithm proposed is called AdaBound. Therefore, AdaBound currently performs better in terms of convergence speed and generalization ability compared with other algorithms.

Although AdaBound performs well in the convergence rate and generalization ability, projection steps in AdaBound produce numbers of computation cost and make training process prohibit when dealing with large-scale problems. To be specific, the projection operator defined as $\Pi_{F,M}$ can be formed as follows:

$$\Pi_{\mathcal{F},M}(\mathbf{y}) = \arg\min_{\mathbf{x}\in\mathcal{F}} \left\| M^{1/2}(\mathbf{x}-\mathbf{y}) \right\|, \tag{1}$$

where $\mathcal{F}$ is a convex feasible set, $\mathbf{x}$ is a decision variable in the feasible domain, and $\mathbf{y}$ is an unknown variable.

Equation (1) shows the high-order calculation method of a projection operation, which brings a lot of calculation cost to the algorithm. The efficiency of algorithms like AdaBound are highly dependent on time and hardware. Therefore, it is necessary to eliminate projection steps of AdaBound in order to improve its efficiency. However, this much needed algorithm has not yet been proposed. For this reason, we propose a projection-free algorithm based on AdaBound, which uses the Frank-Wolfe method to replace high-order projection steps with one-dimensional linear searches.

3. Preliminaries

In this section, we first introduce some notations for convenience. Throughout this paper, we let a boldtype letter, like $\mathbf{x}$, denote a vector. For operative symbol, we let $\mathbf{x}/\mathbf{y}$ denote the element-wise division, $\mathbf{x}^2$ denotes the element-wise square, and $\sqrt{\mathbf{x}}$ denotes the element-wise square root. For the tth iteration, we let $\mathbf{x}_t$ denote the decision vector, f_t denote the cost function, and $x_{t,i}$ denote the ith coordinate of $\mathbf{x}_t$. Moreover, diag $\{\mathbf{x}\}$ denotes a diagonal matrix generated by the elements of $\mathbf{x}$ in order, max $\{\cdot, \cdot\}$ represents element-wise maximum, and $\langle \cdot, \cdot \rangle$ denotes the scalar inner product. In addition, we let R denote the real number set and $\Pi_{\mathscr{F},M}(\cdot)$ denotes the weighted projection operation, where $\mathscr{F}$ is a feasible set and M represents a positive definite matrix.

In this paper, we consider an online convex optimization problem, in which the cost function changes over the time or iteration. If $\mathscr{F} \subset R$ is a convex and compact set, the decision vector $\mathbf{x} \in \mathscr{F}$, and the cost function at time t is f_t; then, we focus on the following optimization objective:

$$\min_{\mathbf{x}\in\mathscr{F}} \sum_{t=1}^{T} f_t(\mathbf{x}). \tag{2}$$

In order to solve the online optimization problem, i.e., Equation (2), an online optimization algorithm is required. In addition, to measure the performance of an online optimization algorithm, one of standard approaches is regret. Moreover, if we let $\mathbf{x}^*$ denote the theoretical optimal solution, then the definition of regret is as follows:

$$R(T) = \sum_{t=1}^{T} f_t(\mathbf{x}_t) - \sum_{t=1}^{T} f_t(\mathbf{x}^*), \tag{3}$$

where $t = 1, \cdots, T$, and $\mathbf{x}^* = \arg\ \min_{\mathbf{x}\in\mathscr{F}} f(\mathbf{x})$.

4. Algorithm Design and Assumptions

In this section, the proposed algorithm design will be firstly introduced in detail. Then, to analyze the convergence of the proposed algorithm, we present some reasonable assumptions.

4.1. Algorithm Design. The input of FWAdaBound is $\mathbf{x}_1 \in \mathscr{F}$, where $\mathscr{F}$ is a convex and compact set. The parameter $\beta_{1t} \in [0, 1)$ and let $\beta_{11} = \beta_1$. Moreover, the parameter $\beta_2 \in [0, 1)$. In addition, the parameters $\alpha, \eta \in (0, 1]$. Let $\mathbf{g}_t$ denote the gradient at time $t \in \{1, \cdots, T\}$; thus, $\mathbf{g}_t = \nabla f_t(\mathbf{x}_t)$. The overall idea of our algorithm is as follows:

At first, we use the first-order momentum of the gradient $\mathbf{u}_t$ to define the sum function for time t: $S_t(\mathbf{x}) = \eta\langle \sum_{\tau=1}^{t} \mathbf{u}_\tau, \mathbf{x}\rangle + \|\mathbf{x} - \mathbf{x}_1\|^2$; then, we use this function to implement one-dimensional linear search $\mathbf{w}_t = \arg\ \min_{\mathbf{x}\in\mathscr{F}} \langle \nabla S_t(\mathbf{x}_t), \mathbf{x}\rangle$, which can accelerate convergence and avoid projection operators, so it is the key of FWAdaBound to reduce the computational cost. Next, we introduce second-order momentum $\mathbf{d}_t$ and use it to generate the dynamic upper bound of learning rate ϖ_t adaptively. Finally, we apply ϖ_t to update the decision variable as $\mathbf{x}_{t+1} = \mathbf{x}_t + \varpi_t e(\mathbf{w}_t - \mathbf{x}_t)$. The specific algorithm is shown in Algorithm 1.

The first-order momentum of the gradient, $\mathbf{u}_t$, is computed by FWAdaBound for time t as follows:

$$\mathbf{u}_t = \beta_{1t}\mathbf{u}_{t-1} + (1 - \beta_{1t})\mathbf{g}_t. \tag{4}$$

The first-order momentum is generated by weighted average of the current gradient and the historical gradient, which speed up convergence rate for optimization algorithms. Next, to implement one-dimensional linear search which replaces of the projection operators, we define the following sum function for time t:

$$S_t(\mathbf{x}) = \eta\left\langle \sum_{\tau=1}^{t} \mathbf{u}_\tau, \mathbf{x} \right\rangle + \|\mathbf{x} - \mathbf{x}_1\|^2. \tag{5}$$

To reduce the computational cost of the projection operation, FWAdaBound searches the feasible variable, $\mathbf{w}_t$, through one-dimensional linear as follows:

$$\mathbf{w}_t = \arg\ \min_{\mathbf{x}\in\mathscr{F}} \langle \nabla S_t(\mathbf{x}_t), \mathbf{x}\rangle. \tag{6}$$

Moreover, to realize the adaptive learning rate, FWAdaBound computes the second-order momentum of the gradient, $\mathbf{d}_t$, for time t as follows:

$$\mathbf{d}_t = \beta_2\mathbf{d}_{t-1} + (1 - \beta_2)\mathbf{g}_t^2. \tag{7}$$

To ensure the convergence of the proposed algorithm, FWAdaBound chooses the bigger value from $\{\mathbf{d}_t, \mathbf{d}_{t-1}\}$ for time t, i.e. $\widehat{\mathbf{d}}_\mathbf{t} = \max\{\mathbf{d}_t, \mathbf{d}_{t-1}\}$. In addition, the diagonal matrix, D_t, based on $\widehat{\mathbf{d}}_\mathbf{t}$ is defined as $D_t = \text{diag}\{\widehat{\mathbf{d}}_t\}$. Next, FWAdaBound generates a dynamic bound for learning rate at time t:

$$\varpi_t = \text{Clip}\left\{\frac{\alpha_t}{\sqrt{D_t}}, \frac{\varpi_{\text{low}}(t)}{\sqrt{t}}, \frac{\varpi_{\text{upp}}(t)}{\sqrt{t}}\right\}, \tag{8}$$

where $\varpi_{\text{low}}(t)$ is the lower bound and $\varpi_{\text{upp}}(t)$ is the upper bound. Therefore, Equation (5) clips the output of $\alpha_t/\sqrt{D_t}$ between the low bound and the upper bound. Finally, FWAdaBound updates the decision variable for time $t + 1$ as follows:

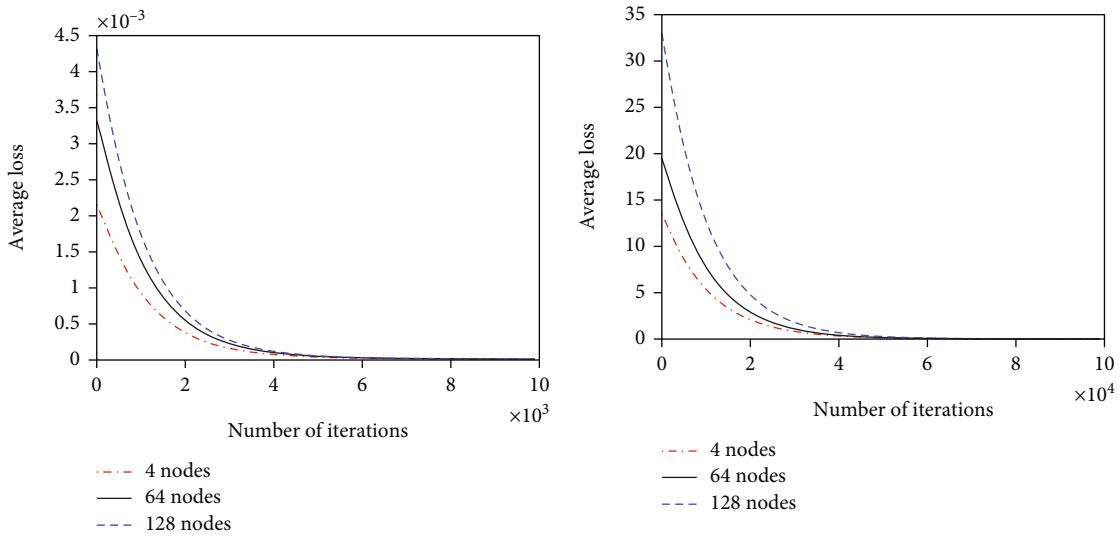
$$\mathbf{x}_{t+1} = \mathbf{x}_t + \varpi_t \bigodot (\mathbf{w}_t - \mathbf{x}_t). \tag{9}$$

Therefore, the design of the proposed algorithm is introduced completely. And we present some following common assumptions, which are the premises of the convergence of the algorithm.

4.2. Assumptions. Next, three assumptions are presented for the proposed algorithm as follows.

Input: $\mathbf{x}_1$
Parameter: $\mathbf{x}_1 \in \mathcal{F}$, and $\beta_{1t} \in [0, 1)$ where $\beta_{11} = \beta_1$, $\beta_2 \in [0, 1)$. Moreover, $\alpha, \eta \in (0, 1]$.
Initially set: $\mathbf{m}_1 = \mathbf{0}$ and $\mathbf{v}_1 = \mathbf{0}$.
Output: $\mathbf{x}_{t+1}$
for $t = 1, 2, 3, \cdots$ **do**
  $t \longleftarrow t + 1$
  Compute gradient of decision variables at time t:
  $\mathbf{g}_t = \nabla f_t(\mathbf{x}_t)$
  Compute the first-order momentum at time t:
  $\mathbf{u}_t = \beta_{1t}\mathbf{u}_{t-1} + (1 - \beta_{1t})\mathbf{g}_t$
  Generate a new sum function:
  $S_t(\mathbf{x}) = \eta\sum_{\tau=1}^{t} \boxtimes \mathbf{u}_\tau, \mathbf{x} + \|\mathbf{x} - \mathbf{x}_1\|^2$
  Search $\mathbf{w}_t$ by one-dimensional linearly:
  $\mathbf{w}_t = \arg\ \min_{\mathbf{x}\in\mathcal{F}} \nabla S_t(\mathbf{x}_t), \mathbf{x}$
  Compute the second-order momentum at time t:
  $\mathbf{d}_t = \beta_2\mathbf{d}_{t-1} + (1 - \beta_2)\mathbf{g}_t^2$
  Select a bigger value for the second-order momentum:
  $\widehat{\mathbf{d}}_t = \max\{\mathbf{d}_t, \mathbf{d}_{t-1}\}$ and $D_t = \text{diag}\{\widehat{\mathbf{d}}_t\}$
  Compute the dynamic bound for learning rate at time t:
  $\varpi_t = \text{Clip}\{(\alpha_t/\sqrt{D_t}), (\varpi_{\text{low}}(t)/\sqrt{t}), (\varpi_{\text{upp}}(t)/\sqrt{t})\}$
  Update the decision variables for time $t + 1$:
  $\mathbf{x}_{t+1} = \mathbf{x}_t + \varpi_t \mathbf{e}(\mathbf{w}_t - \mathbf{x}_t)$
end for
return $\mathbf{x}_{t+1}$

Algorithm 1: FWAdaBound



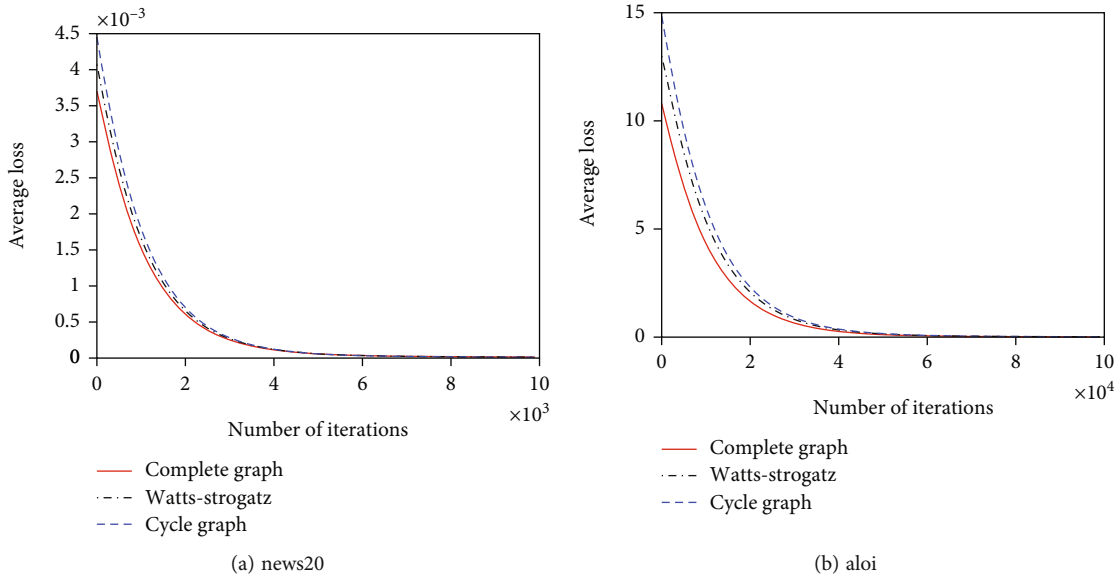
(a) news20 (b) aloi

Figure 1: Comparison of the relationship between the average loss and the running time of each algorithm.

Assumption 1. The constraint set $\mathcal{F}$ is convex and compact. Moreover, the set $\mathcal{F}$ is bounded, i.e., $\|\mathbf{x} - \mathbf{y}\|_\infty \le B_\infty$ for all $\mathbf{x}, \mathbf{y} \in \mathcal{F}$, where $B_\infty > 0$.

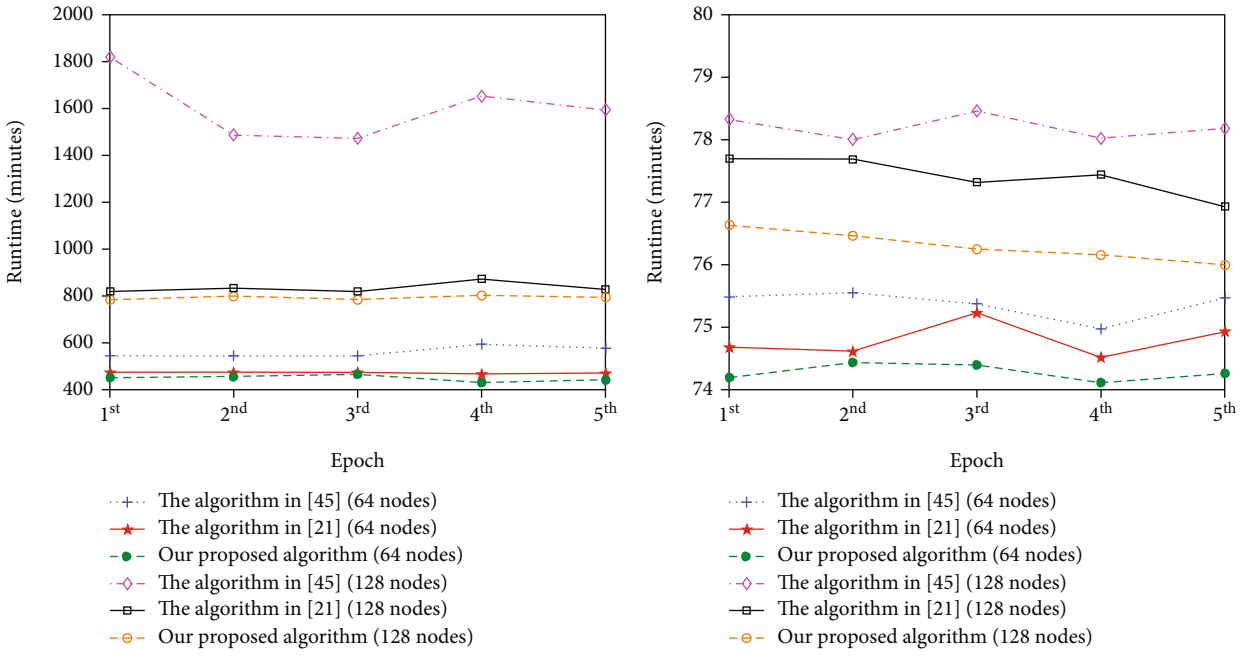
Assumption 2. The cost function f_t of FWAdaBound is convex and differentiable on $\mathcal{F}$ for all $t \in \{1, \cdots, T\}$. In addition, all the cost functions, $\{f_1, \cdots, f_t\}$, are Lipschitz functions with L constant, where $L > 0$.

Assumption 3. The gradient of decision variable $\mathbf{x}_t$ is bounded for all $t \in \{1, 2, \cdots, T\}$ over $\mathcal{F}$, i.e., $\mathbf{g}_t = \|\nabla f_t(\mathbf{x}_t)\| \le G_\infty$, where $G_\infty > 0$.



(a) news20 (b) aloi

Figure 2: Comparison of the relationship between the training accuracy and the running time of each algorithm.



(a) news20 (b) aloi

Figure 3: Comparison of the relationship between the test accuracy and the running time of each algorithm.

Assumption 1 is one of the most basic assumptions of projection-free method, and almost all projection-free related articles use it, such as these classic articles [19, 20]. Assumption 2–3 are supposed commonly and reasonably for analyzing convergence of optimization algorithms such as these research works [14–16]. Next, we present the convergence analysis of the proposed algorithms based on Assumptions 1–3.

In many research works [14–16], Assumptions 1–3 were supposed commonly and reasonably for analyzing convergence of proposed algorithms. Next, we present the convergence analysis of the proposed algorithms based on Assumptions 1–3.

5. Convergence Analysis

We first introduce the following definitions as the beginning of this section. Moreover, the introduced definitions are standard and common in convex optimization.

$$|f(\mathbf{x}) - f(\mathbf{y})| \leq L\|\mathbf{x} - \mathbf{y}\|, \tag{10}$$

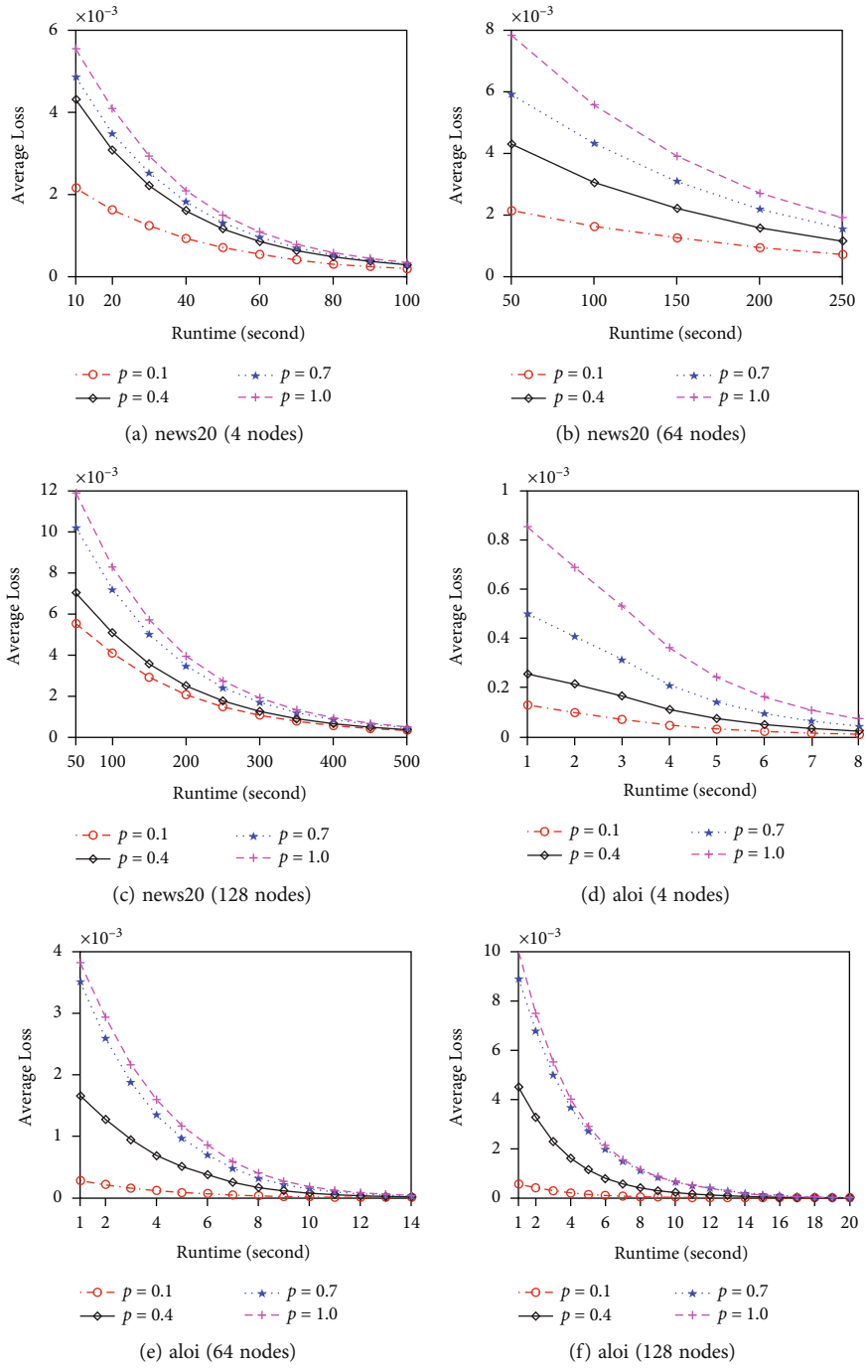


Figure 4: Comparison of the relationship between the perplexity and the running time of each algorithm. Lower is better.

Definition 4. A function $f : \mathcal{F} \mapsto R$ is called L-Lipchitz if for any two points $\mathbf{x}, \mathbf{y} \in \mathcal{F}$ we have

where L is a positive constant.

Definition 5. A function $f : \mathcal{F} \mapsto R$ is convex and differentiable if for all $\mathbf{x}, \mathbf{y} \in \mathcal{F}$, we have

$$f(\mathbf{x}) - f(\mathbf{y}) \leq \nabla f(\mathbf{x})^T (\mathbf{x} - \mathbf{y}). \tag{11}$$

Definition 6. Let $f : \mathcal{F} \mapsto R$ be an arbitrary convex function. Then, the function f is also called μ-smooth if for any two points $\mathbf{x}, \mathbf{y} \in \mathcal{F}$, we have

$$f(\mathbf{x}) - f(\mathbf{y}) \geq \nabla f(\mathbf{x})^T (\mathbf{x} - \mathbf{y}) - \frac{\mu}{2} \|\mathbf{y} - \mathbf{x}\|^2, \tag{12}$$

where $\mu > 0$.

Definition 7. Let $f : \mathcal{F} \mapsto R$ be an arbitrary convex function. Then, the function f is δ-strongly convex if for all $\mathbf{x}, \mathbf{y} \in \mathcal{F}$,

we have

$$f(\mathbf{x}) - f(\mathbf{y}) \leq \nabla f(\mathbf{x})^T(\mathbf{x}-\mathbf{y}) - \frac{\delta}{2}\|\mathbf{y}-\mathbf{x}\|^2, \tag{13}$$

where $\delta > 0$.

In addition, if a function f is δ-strongly convex and let $\mathbf{x}^* = \arg\ \min_{\mathbf{x}\in\mathcal{F}} f(\mathbf{x})$, then we have

$$f(\mathbf{x}) - f(\mathbf{x}^*) \geq \frac{\delta}{2}\|\mathbf{x}-\mathbf{x}^*\|^2. \tag{14}$$

Moreover, from Definition 6, we obtain the following equivalent relation:

$$\|\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})\| \leq \mu\|\mathbf{x}-\mathbf{y}\|, \tag{15}$$

where $\mathbf{x}, \mathbf{y} \in \mathcal{F}$.

In order to simplify the process of convergence analysis, we define some intermediate variables. Let $\mathbf{x}_t^* = \arg\ \min_{\mathbf{x}\in\mathcal{F}} S_t(\mathbf{x})$ for any $t \in \{1, \cdots, T\}$. Moreover, we define $S_0(\mathbf{x}) = \|\mathbf{x} - \mathbf{x}_1\|^2$ at time $t = 0$. In addition, we present the following relation for S_t:

$$z_t(\mathbf{x}) = S_t(\mathbf{x}) - S_t(\mathbf{x}_t^*). \tag{16}$$

When $\mathbf{x} = \mathbf{x}_t$, for brevity, denoting $z_t = z_t(\mathbf{x}_t)$. Next, we present the following Lemma 8 to bound z_{t+1}.

Lemma 8. *If Assumptions 1–3 are satisfied, and variables* $\{\mathbf{x}_t\}, \{\mathbf{u}_t\}$, *and* $\{\mathbf{d}_t\}$ *are generated by Algorithm 1 for* $t \in \{1, \cdots, T\}$, *and* $\beta_{1t} = \beta_1\lambda^{t-1} \leq \beta_{1(t-1)} \leq \beta_1$, *where* $\lambda \in (0, 1]$. *In addition, suppose that* $0 \leq \varpi_{low}(t) \leq \varpi_{low}(t+1)$, *and* $0 \leq \varpi_{upp}(t+1) \leq \varpi_{upp}(t)$, *denoting* $B_\infty = \varpi_{upp}(1)$ *and* $C_\infty = \varpi_{low}(1)$. *Then, we have*

$$z_{t+1} \leq \left(1 - \frac{B_\infty^2}{\sqrt{t}}\right) z_t + \frac{n\eta G_\infty}{1-\beta_1}\sqrt{z_{t+1}} + \frac{B_\infty^4}{t}. \tag{17}$$

Proof. By Definition 6, we know that $S_t(\mathbf{x})$ is a 2-smooth function. Moreover, using the definition of $z_t(\mathbf{x})$ (i.e., Equation (16)) and $\mathbf{x}_t$, we obtain the following relation:

$$z_t(\mathbf{x}_{t+1}) = S_t(\mathbf{x}_{t+1}) - S_t(\mathbf{x}_t^*) = S_t(\mathbf{x}_t + \varpi_t \mathbf{e}(\mathbf{w}_t - \mathbf{x}_t)) - S_t(\mathbf{x}_t^*). \tag{18}$$

□

From the bounds of $\varpi_{\text{low}}(t)$ and $\varpi_{\text{upp}}(t)$, we have the following:

$$z_t(\mathbf{x}_{t+1}) \leq S_t\left(\mathbf{x}_t + \frac{\varpi_{\text{upp}}(1)}{\sqrt{t}}\mathbf{e}(\mathbf{w}_t - \mathbf{x}_t)\right) - S_t(\mathbf{x}_t^*) \leq S_t\left(\mathbf{x}_t + \frac{B_\infty}{\sqrt{t}}\mathbf{e}(\mathbf{w}_t - \mathbf{x}_t)\right) - S_t(\mathbf{x}_t^*). \tag{19}$$

In addition, from the Definition 7 and the strong-convexity of $S_t(\mathbf{x})$, we obtain the following:

$$z_t(\mathbf{x}_{t+1}) \leq S_t(\mathbf{x}_t) - S_t(\mathbf{x}_t^*) + \frac{B_\infty^2}{t}\|\mathbf{w}_t - \mathbf{x}_t\|^2 + \frac{B_\infty^2}{\sqrt{t}}\langle \nabla S_t(\mathbf{x}_t), (\mathbf{w}_t - \mathbf{x}_t)\rangle. \tag{20}$$

From the definition of $\mathbf{w}_t$, we attain the following relation:

$$\langle \nabla S_t(\mathbf{x}_t), \mathbf{w}_t\rangle \leq \langle \nabla S_t(\mathbf{x}_t), \mathbf{x}_t^*\rangle. \tag{21}$$

Moreover, from Equation (21), we have the following:

$$\langle \nabla S_t(\mathbf{x}_t), (\mathbf{w}_t - \mathbf{x}_t)\rangle \leq \langle \nabla S_t(\mathbf{x}_t), (\mathbf{x}_t^* - \mathbf{x}_t)\rangle. \tag{22}$$

Plugging Equation (22) into Equation (20), we attain the following relation:

$$z_t(\mathbf{x}_{t+1}) \leq S_t(\mathbf{x}_t) - S_t(\mathbf{x}_t^*) + \frac{B_\infty^2}{t}\|\mathbf{w}_t - \mathbf{x}_t\|^2 + \frac{B_\infty^2}{\sqrt{t}}\langle \nabla S_t(\mathbf{x}_t), (\mathbf{x}_t^* - \mathbf{x}_t)\rangle. \tag{23}$$

According to Definition 5 and the convexity of $S_t(\mathbf{x})$, we obtain the following relation:

$$\langle \nabla S_t(\mathbf{x}_t), (\mathbf{x}_t^* - \mathbf{x}_t)\rangle \leq S_t(\mathbf{x}^*) - S_t(\mathbf{x}_t). \tag{24}$$

Furthermore, plugging Equation (24) into Equation (23), we get the following relation:

$$\begin{aligned} z_t(\mathbf{x}_{t+1}) &\leq S_t(\mathbf{x}_t) - S_t(\mathbf{x}_t^*) + \frac{B_\infty^2}{t}\|\mathbf{w}_t - \mathbf{x}_t\|^2 + \frac{B_\infty^2}{\sqrt{t}}(S_t(\mathbf{x}^*) - S_t(\mathbf{x}_t)) \\ &\leq \left(1 - \frac{B_\infty^2}{\sqrt{t}}\right)(S_t(\mathbf{x}_t) - S_t(\mathbf{x}_t^*)) + \frac{B_\infty^2}{t}\|\mathbf{w}_t - \mathbf{x}_t\|^2. \end{aligned} \tag{25}$$

Next, we consider the term $z_t(\mathbf{x}_{t+1})$ in Equation (25). By the definition of $z_t(\mathbf{x})$, we first obtain the following relation:

$$z_{t+1}(\mathbf{x}_{t+1}) \leq S_{t+1}(\mathbf{x}_{t+1}) - S_{t+1}(\mathbf{x}_{t+1}^*). \tag{26}$$

Then, transforming Equation (26), and we attain the following relation:

$$z_{t+1}(\mathbf{x}_{t+1}) \leq S_t(\mathbf{x}_{t+1}) - S_t(\mathbf{x}_{t+1}^*) + S_{t+1}(\mathbf{x}_{t+1}) - S_t(\mathbf{x}_{t+1}) + S_{t+1}(\mathbf{x}_{t+1}^*) - S_t(\mathbf{x}_{t+1}^*). \tag{27}$$

In addition, due to the fact that $\mathbf{x}_t^* = \arg\ \min_{\mathbf{x}\in\mathcal{F}} S_t(\mathbf{x})$, we have $S_t(\mathbf{x}_t^*) \leq S_t(\mathbf{x}_{t+1}^*)$. For this reason, we obtain the following relation from Equation (27):

$$\begin{aligned} z_{t+1}(\mathbf{x}_{t+1}) &\leq S_t(\mathbf{x}_{t+1}) - S_t(\mathbf{x}_t^*) + S_{t+1}(\mathbf{x}_{t+1}) - S_t(\mathbf{x}_{t+1}) + S_{t+1}(\mathbf{x}_{t+1}^*) - S_t(\mathbf{x}_{t+1}^*) \\ &= z_t(\mathbf{x}_{t+1}) + S_{t+1}(\mathbf{x}_{t+1}) - S_t(\mathbf{x}_{t+1}) + S_{t+1}(\mathbf{x}_{t+1}^*) - S_t(\mathbf{x}_{t+1}^*). \end{aligned} \tag{28}$$

Next, we consider the terms $S_{t+1}(\mathbf{x}_{t+1}) - S_t(\mathbf{x}_{t+1})$ and $S_{t+1}(\mathbf{x}_{t+1}^*) - S_t(\mathbf{x}_{t+1}^*)$ in Equation (28). From the definition

of $S_t(\mathbf{x})$, we have the following relation:

$$S_t(\mathbf{x}) - S_t(\mathbf{x}) = \eta \sum_{\tau=1}^{t+1} \mathbf{u}_\tau^T \mathbf{x} + \|\mathbf{x} - \mathbf{x}_1\|^2 - \eta \sum_{\tau=1}^{t} \mathbf{u}_\tau^T \mathbf{x} - \|\mathbf{x} - \mathbf{x}_1\|^2 = \eta \mathbf{u}_{t+1}^T \mathbf{x}. \tag{29}$$

Let $\mathbf{x} = \mathbf{x}_{t+1}$ in Equation (29), we obtain

$$S_t(\mathbf{x}_{t+1}) - S_t(\mathbf{x}_{t+1}) = \eta \mathbf{u}_{t+1}^T \mathbf{x}_{t+1}. \tag{30}$$

In addition, let $\mathbf{x} = \mathbf{x}_{t+1}^*$ in Equationss (29), we attain

$$S_t(\mathbf{x}_{t+1}^*) - S_t(\mathbf{x}_{t+1}^*) = \eta \mathbf{u}_{t+1}^T \mathbf{x}_{t+1}^*. \tag{31}$$

Combining Equations (28), (30), and (31), we have the following:

$$\begin{aligned} z_{t+1}(\mathbf{x}_{t+1}) &\le z_t(\mathbf{x}_{t+1}) + \eta \mathbf{u}_{t+1}^T \mathbf{x}_{t+1} + \eta \mathbf{u}_{t+1}^T \mathbf{x}_{t+1}^* = z_t(\mathbf{x}_{t+1}) + \eta \mathbf{u}_{t+1}^T (\mathbf{x}_{t+1} - \mathbf{x}_{t+1}^*) \\ &\le z_t(\mathbf{x}_{t+1}) + \eta \left\| \mathbf{u}_{t+1}^T \right\| \|\mathbf{x}_{t+1} - \mathbf{x}_{t+1}^*\|. \end{aligned} \tag{32}$$

The second inequality in Equation (32) follows from Cauchy-Schwarz inequality. Besides, plugging Equation (23) into Equation (32), we obtain

$$z_{t+1}(\mathbf{x}_{t+1}) \le \left(1 - \frac{B_\infty^2}{\sqrt{t}}\right)(S_t(\mathbf{x}_t) - S_t(\mathbf{x}_t^*)) + \frac{B_\infty^2}{t}\|\mathbf{w}_t - \mathbf{x}_t\|^2 + \eta \left\| \mathbf{u}_{t+1}^T \right\| \|\mathbf{x}_{t+1} - \mathbf{x}_{t+1}^*\|. \tag{33}$$

Since $z_t(\mathbf{x}_t) = S_t(\mathbf{x}_t) - S_t(\mathbf{x}_t^*)$, we attain the following relation from Equation ((33))

$$\begin{aligned} z_{t+1}(\mathbf{x}_{t+1}) &\le \left(1 - \frac{B_\infty^2}{\sqrt{t}}\right) z_t(\mathbf{x}_t) + \frac{B_\infty^2}{t}\|\mathbf{w}_t - \mathbf{x}_t\|^2 + \eta \left\| \mathbf{u}_{t+1}^T \right\| \|\mathbf{x}_{t+1} - \mathbf{x}_{t+1}^*\| \\ &\le \left(1 - \frac{B_\infty^2}{\sqrt{t}}\right) z_t + \frac{B_\infty^2}{t}\|\mathbf{w}_t - \mathbf{x}_t\|^2 + \eta \left\| \mathbf{u}_{t+1}^T \right\| \|\mathbf{x}_{t+1} - \mathbf{x}_{t+1}^*\|. \end{aligned} \tag{34}$$

The second inequality in Equation (34) follows the definition of $z_t = z_t(\mathbf{x}_t)$.

Before estimating the bound of $z_{t+1}(\mathbf{x}_{t+1})$, we consider the bound of $\|\mathbf{u}_{t+1}\|$. To this end, applying the recursive algorithm on $\mathbf{u}_t$, we have

$$\mathbf{u}_t = \sum_{j=1}^{t} \left(1 - \beta_{1j}\right) \prod_{k=1}^{t-j} \beta_{1(t-k+1)} \mathbf{g}_j. \tag{35}$$

From Assumption 3 and Equation (35), we attain

$$\begin{aligned} \|\mathbf{u}_t\| &\le \sum_{j=1}^{t} \left(1 - \beta_{1j}\right) \prod_{k=1}^{t-j} \beta_{1(t-k+1)} \left(\sum_{\sigma=1}^{n} g_{j,\sigma} \right) \\ &\le nG_\infty \sum_{j=1}^{t} \left(1 - \beta_{1j}\right) \prod_{k=1}^{t-j} \beta_{1(t-k+1)} \le nG_\infty \sum_{j=1}^{t} \prod_{k=1}^{t-j} \beta_{1(t-k+1)}. \end{aligned} \tag{36}$$

In addition, because $\beta_{1t} = \beta_1 \lambda^{t-1}$, we obtain the following relation from Equation (36):

$$\|\mathbf{u}_t\| \le nG_\infty \sum_{j=1}^{t} \beta_1^{t-j} \le \frac{nG_\infty}{1 - \beta_1}. \tag{37}$$

Plugging Equation (37) into Equation (34), we attain

$$z_{t+1} \le \left(1 - \frac{B_\infty^2}{\sqrt{t}}\right) z_t + \frac{B_\infty^2}{t}\|\mathbf{w}_t - \mathbf{x}_t\|^2 + \frac{n\eta G_\infty}{1 - \beta_1}\|\mathbf{x}_{t+1} - \mathbf{x}_{t+1}^*\|. \tag{38}$$

Since $\mathbf{w}_t = \arg\min_{\mathbf{x} \in \mathscr{F}} \langle \nabla S_t(\mathbf{x}_t), \mathbf{x} \rangle$, we have $\mathbf{w}_t \in \mathscr{F}$. Moreover, from Assumption 1, we have

$$\|\mathbf{w}_t - \mathbf{x}_t\| \le B_\infty. \tag{39}$$

Hence, combining Equations (38) and (40), we obtain

$$z_{t+1} \le \left(1 - \frac{B_\infty^2}{\sqrt{t}}\right) z_t + \frac{n\eta G_\infty}{1 - \beta_1}\|\mathbf{x}_{t+1} - \mathbf{x}_{t+1}^*\| + \frac{B_\infty^4}{t}. \tag{40}$$

Applying $\delta = 2$ on Definition 7, we attain that the function $S_t(\mathbf{x})$ is 2-strongly convex. In addition, from Equation (14), we have

$$\|\mathbf{x}_{t+1} - \mathbf{x}_{t+1}^*\| \le \sqrt{S_{t+1}(\mathbf{x}_{t+1}) - S_{t+1}(\mathbf{x}_{t+1}^*)} = \sqrt{z_{t+1}}. \tag{41}$$

Moreover, substituting Equation (41) into Equation (40), and we can obtain the following:

$$z_{t+1} \le \left(1 - \frac{B_\infty^2}{\sqrt{t}}\right) z_t + \frac{n\eta G_\infty}{1 - \beta_1}\sqrt{z_{t+1}} + \frac{B_\infty^4}{t}. \tag{42}$$

Therefore, the proof of Lemma 8 is completed.

Now, we get the iterative relations between z_t and z_{t+1} from Lemma 8. In order to attain the final bound of z_t, we should present the following two lemmas. First of all, we introduce the first lemma of the two lemmas.

Lemma 9. *For all $t = 1, 2, \cdots$, we can obtain the following relation:*

$$\frac{1}{\sqrt{t}}\left(1 - \frac{1}{2\sqrt{t}}\right) \le \frac{1}{\sqrt{t+1}}. \tag{43}$$

Proof. We first square both sides of Equation (43) and take

the difference; then, we have the following relation:

$$\begin{aligned}\left(\frac{1}{\sqrt{t}}\left(1-\frac{1}{2\sqrt{t}}\right)\right)^2-\left(\frac{1}{\sqrt{t+1}}\right)^2&=\frac{1}{t}-\frac{1}{t\sqrt{t}}+\frac{1}{4t^2}-\frac{1}{t+1}\\&=\left(\frac{1}{t}-\frac{1}{t\sqrt{t}}+\frac{1}{4t^2}-\frac{1}{t+1}\right)\frac{t(t+1)}{t(t+1)}\\&=\frac{t+1+(t+1/4t)-t+1/\sqrt{t}-t}{t(t+1)}\\&=\frac{t+(t+1/4)-(t+1)\sqrt{t}}{t^2(t+1)}=\frac{(5t+1)-4\sqrt{t}(t+1)\sqrt{t}}{t^2(t+1)}.\end{aligned}\tag{44}$$

□

Observing terms $(5t+1)$ and $4\sqrt{t}(t+1)\sqrt{t}$, we can know that they all increase with t, and the latter grows faster than the former. Therefore, we can further attain the following relation from Equation (44):

$$\left(\frac{1}{\sqrt{t}}\left(1-\frac{1}{2\sqrt{t}}\right)\right)^2-\left(\frac{1}{\sqrt{t+1}}\right)^2=\frac{(5t+1)-4\sqrt{t}(t+1)\sqrt{t}}{t^2(t+1)}\le 0.\tag{45}$$

In addition, combining Equations (44) and (43), we obtain the following relation:

$$\left(\frac{1}{\sqrt{t}}\left(1-\frac{1}{2\sqrt{t}}\right)\right)^2\le\left(\frac{1}{\sqrt{t+1}}\right)^2.\tag{46}$$

From Equation (46), we have the following relation:

$$\frac{1}{\sqrt{t}}\left(1-\frac{1}{2\sqrt{t}}\right)\le\frac{1}{\sqrt{t+1}}.\tag{47}$$

Therefore, the proof of Lemma 9 is completed.

Next, we introduce the last lemma about the final bound of z_t as follows.

Lemma 10. *If Assumptions 1–3 are satisfied, and variables $\{\mathbf{x}_t\},\{\mathbf{u}_t\},\{\mathbf{d}_t\}$ are generated by Algorithm 4.1 for $t=\{1,\cdots,T\}$, and $\beta_{1t}=\beta_1\lambda^{t-1}\le\beta_{1(t-1)}\le\beta_1$, where $\lambda\in(0,1]$. In addition, suppose that $0\le\varpi_{low}(t)\le\varpi_{low}(t+1)$, and $0\le\varpi_{upp}(t+1)\le\varpi_{upp}(t)$. Denoting $B_\infty=\varpi_{upp}(1)$, and $C_\infty=\varpi_{low}(1)$. Moreover, as the parameters n, η, and β_1 are chosen such that $(n\eta G_\infty/(1-\beta_1)1-\beta_1)\sqrt{z_{t+1}}\le(3B_\infty^4-2B_\infty^2)/t$ by Algorithm 1, we then have*

$$z_{t+1}\le\frac{4B_\infty^2}{\sqrt{t+1}}.\tag{48}$$

Proof. By Equation (42), we have the following relation:

$$z_{t+1}\le\left(1-\frac{B_\infty^2}{\sqrt{t}}\right)z_t+\frac{4B_\infty^4-2B_\infty^2}{t}.\tag{49}$$

□

Now, we can use mathematical induction to get the bound of z_t. First, when $t=1$, from definition of z_t, we attain

$$z_1\le S_1(\mathbf{x}_1)-S_1(\mathbf{x}_1^*)\le\|\mathbf{x}_1-\mathbf{x}_1\|^2-\|\mathbf{x}_1^*-\mathbf{x}_1\|^2=-\|\mathbf{x}_1^*-\mathbf{x}_1\|^2\le 4B_\infty^2.\tag{50}$$

Therefore, the base of mathematical induction is true for $t=1$. Second, supposing that the mathematical induction is also true for t, and we present that it also true for $t+1$ as follows:

$$\begin{aligned}z_{t+1}&\le\left(1-\frac{B_\infty^2}{\sqrt{t}}\right)\frac{4B_\infty^2}{\sqrt{t}}+\frac{4B_\infty^4-2B_\infty^2}{t}\le\frac{4B_\infty^2}{\sqrt{t}}-\frac{4B_\infty^4}{t}+\frac{4B_\infty^4}{t}-\frac{2B_\infty^2}{t}\\&\le\frac{4B_\infty^2}{\sqrt{t}}\left(1-\frac{1}{2\sqrt{t}}\right).\end{aligned}\tag{51}$$

Applying Lemma 9 into Equation (51), we obtain

$$z_{t+1}\le\frac{4B_\infty^2}{\sqrt{t+1}}.\tag{52}$$

Therefore, the proof of Lemma 10 is completed.

Next, we present the following result to attain the bound of $R(T)$.

Theorem 11. *If the Assumptions 1–3 are satisfied. Moreover, the sequences $\mathbf{u}_t$, $\mathbf{w}_t$, and $\mathbf{d}_t$ are all generated by our proposed algorithm, which $t\in\{1,2,\cdots,T\}$. Then, we obtain that*

$$R(T)\le\frac{8LB_\infty}{3}T^{3/4}+\frac{B_\infty^2}{\eta(1-\beta_1)\beta_{1T}}.\tag{53}$$

Proof. From Lemma 10 and Equation (41), we have the following:

$$\|\mathbf{x}_t-\mathbf{x}_t^*\|\le\sqrt{z_t}\le\sqrt{\frac{4B_\infty^2}{\sqrt{t}}}=2B_\infty t^{-1/4}.\tag{54}$$

□

Summing over Equation (54) for $t=1,\cdots,T$, we attain the following relation:

$$\sum_{t=1}^{T}\|\mathbf{x}_t-\mathbf{x}_t^*\|\le\sum_{t=1}^{T}2B_\infty t^{-1/4}\le\frac{8B_\infty}{3}T^{3/4}.\tag{55}$$

By Assumption 2, we know that f_t is a Lipschitz function. In addition, applying Definition 4, we have

$$|f_t(\mathbf{x}_t)-f_t(\mathbf{x}_t^*)|\le L\|\mathbf{x}_t-\mathbf{x}_t^*\|.\tag{56}$$

In addition, combining Equation (55) and (56), we obtain

$$\sum_{t=1}^{T}|f_t(\mathbf{x}_t)-f_t(\mathbf{x}_t^*)|\le\sum_{t=1}^{T}L\|\mathbf{x}_t-\mathbf{x}_t^*\|\le\frac{8LB_\infty}{3}T^{3/4}.\tag{57}$$

Moreover, from the definition of $R(T)$, we have

$$R(T) = \sum_{t=1}^{T}[f_t(\mathbf{x}_t) - f_t(\mathbf{x}^*)] = \sum_{t=1}^{T}[f_t(\mathbf{x}_t) - f_t(\mathbf{x}_t^*)] + \sum_{t=1}^{T}[f_t(\mathbf{x}_t^*) - f_t(\mathbf{x}^*)]. \tag{58}$$

Since $S_0(\mathbf{x}) = \|\mathbf{x} - \mathbf{x}_1\|^2$, we attain the following relation:

$$\begin{aligned} S_T(\mathbf{x}_t^*) - S_T(\mathbf{x}^*) &\le \left[\eta\sum_{t=1}^{T}\mathbf{u}_t\mathbf{x}_t^* + \|\mathbf{x}_t^* - \mathbf{x}_1\|^2\right] - \left[\eta\sum_{t=1}^{T}\mathbf{u}_t\mathbf{x}_t^* + \|\mathbf{x}^* - \mathbf{x}_1\|^2\right] \\ &\le \eta\sum_{t=1}^{T}\mathbf{u}_t(\mathbf{x}_t^* - \mathbf{x}^*) + \|\mathbf{x}_t^* - \mathbf{x}_1\|^2 - \|\mathbf{x}^* - \mathbf{x}_1\|^2 \le 0. \end{aligned} \tag{59}$$

Since $\mathbf{x}^*, \mathbf{x}_1 \in \mathcal{F}$, and by Assumption 1, we have $\|\mathbf{x}^* - \mathbf{x}_1\| \le B_\infty$. Moreover, according to Equation (59), we obtain the following relation:

$$\sum_{t=1}^{T}\mathbf{u}_t(\mathbf{x}_t^* - \mathbf{x}^*) \le \frac{1}{\eta}\left[\|\mathbf{x}^* - \mathbf{x}_1\|^2 - \|\mathbf{x}_t^* - \mathbf{x}_1\|^2\right] \le \frac{B_\infty^2}{\eta}. \tag{60}$$

Then, combining Equations (35) and (60), we have the following relation:

$$\begin{aligned} \sum_{t=1}^{T}\mathbf{u}_t(\mathbf{x}_t^* - \mathbf{x}^*) &= \sum_{t=1}^{T}\left[\sum_{j=1}^{t}\left(1-\beta_{1j}\right)\prod_{k=1}^{t-j}\beta_{1(t-k+1)}\mathbf{g}_j\right] \times (\mathbf{x}_t^* - \mathbf{x}^*) \\ &\ge \sum_{t=1}^{T}\left[\sum_{j=1}^{t}\left(1-\beta_{1j}\right)\beta_{1t}\mathbf{g}_j\right](\mathbf{x}_t^* - \mathbf{x}^*) \\ &\ge (1-\beta_1)\beta_{1T}\sum_{t=1}^{T}\mathbf{g}_t(\mathbf{x}_t^* - \mathbf{x}^*). \end{aligned} \tag{61}$$

Therefore, from Equations (60) and (61), we attain the following relation:

$$\sum_{t=1}^{T}\mathbf{g}_t(\mathbf{x}_t^* - \mathbf{x}^*) \le \frac{B_\infty^2}{\eta(1-\beta_1)\beta_{1T}}. \tag{62}$$

Applying Definition 5 and Equation (58), we obtain

$$R(T) \le \sum_{t=1}^{T}|f_t(\mathbf{x}_t) - f_t(\mathbf{x}_t^*)| + \sum_{t=1}^{T}\mathbf{g}_t(\mathbf{x}_t^* - \mathbf{x}^*). \tag{63}$$

Finally, substituting Equations (57) and (62) into Equation (63), we have

$$R(T) \le \frac{8LB_\infty}{3}T^{3/4} + \frac{B_\infty^2}{\eta(1-\beta_1)\beta_{1T}}. \tag{64}$$

Therefore, the proof of Theorem 11 is completed.

From Theorem 11, we get that $\lim_{T\longrightarrow\infty}R(T)/T = 0$, which indicates that our proposed algorithm is convergent. In addition, by Equation (64), we know that the regret bound of our proposed algorithm is $O(T^{3/4})$.

Remark 12. This work is closed to the previous works [14–16]. Our regret bound $O(T^{3/4})$ is worse than the regret bound $O(\sqrt{T})$, which is obtained by [15, 16], but the number of iterations is increased within the same amount of time due to the lower computational cost per iteration. Hence, the overall convergent rate of the proposed algorithm is faster than the Adam-type algorithms such as Adam [14] and AdaBound [16].

Next, we will validate the performance of our proposed algorithm by simulation experiments.

6. Experiments

Our algorithm is mainly used in the field of multimedia communication. Specifically, through modeling and analysis, it is proved that the algorithm can reduce the calculation and storage cost of text, voice, picture, and other information in multimedia information transmission.

So, in this section, we, respectively, apply and compare the proposed algorithm on different dataset to validate the convergence and performance of the proposed algorithm. One application is the image classification on CIFAR-10 [21] dataset, and another one is the language modeling on the Penn Treebank dataset [22]. The experiments are executed on the equipment with 1080Ti GPU and CUDA 0.4.0, and written in Python 3.7 with Torch 1.0.1 framework. The details of experiment settings and results are described in the content below.

6.1. Experiment Settings. The CIFAR-10 is a famous and standard dataset for image classification, which consists of 60,000 32×32 color images in 10 classes, and with 6,000 images per class. Moreover, the dataset has 50,000 training images and 10,000 test images, respectively. We use deep models, ResNet-34 and DenseNet-121, to finish the classification tasks on CIFAR-10. The model ResNet-34, a popular model in deep learning, is a deep residual networks with 34 layers. In addition, DenseNet-121 is a dense convolutional network with 121 layers, in which each layer accepts all preceding layers as its additional input.

The Penn Treebank is a popular and classical dataset for language modeling. The corpus of this dataset comes from the Wall Street Journal. Moreover, this dataset contains 2,499 articles with a total of 1M words. In this experiments, we use three LSTM models, including 1-layer, 2-layer and 3-layer, to train this dataset.

In our experiments, we compare our proposed algorithm, FWAdaBound, with classical and latest proposed algorithms including OGD, Adam [14], and AdaBound [16]. Furthermore, the setting of parameters of all algorithms are shown as follows.

(i) OGD, the initial step size α is chosen from

$$\{1e-2, 5e-3, 1e-3, 5e-4, 1e-4\} \quad (65)$$

(ii) Adam [14], with $\beta_1 = 0.9, \beta_2 = 0.999, \alpha_t = \alpha/\sqrt{t}$, in which the initial step size α is chosen from

$$\{1e-2, 5e-3, 1e-3, 5e-4, 1e-4\} \quad (66)$$

Moreover, we use for the perturbation value $\varepsilon = 1e-8$.

(iii) AdaBound [16], with $\beta_1 = 0.9, \beta_2 = 0.999, \alpha_t = \alpha/\sqrt{t}$, in which the initial step size α is chosen from

$$\{1e-2, 5e-3, 1e-3, 5e-4, 1e-4\} \quad (67)$$

Moreover, $\varepsilon = 1e-8$.

(iv) FWAdaBound, our proposed algorithm, we directly applied the default hyperparameters of AdaBound (a learning rate of 0.001, $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\varepsilon = 1e-8$)

Next, we show the results of our experiments on CIFAR-10 and present the related analysis of the experimental results.

6.2. Experiment Results and Analysis

6.2.1. Image Classification. In the first experiment, we run the algorithms on CIFAR-10 for 200 epochs and measure the relationship between running time and average loss. It can be concluded from the first experiment that with the same epoch number, FWAdaBound spends the shortest time to complete the iterative task. It also confirms that FWAdaBound takes the least computation cost among all the experimental algorithms. The reason is that FWAdaBound uses linear search instead of the high-order projection steps in Adam and AdaBound. Moreover, Figure 1 shows the results for each algorithm on both ResNet-34 and DenseNet-121. In the part (a) of Figure 1, the average loss of our proposed algorithm FWAdaBound reaches the expected stable value in the shortest time in model ResNet-34. Similarly, in part (b) of Figure 1, FWAdaBound also takes the least time to reduce the average loss in model DenseNet-121. This suggests that with the same number of epochs, FWAdaBound spends the least time to complete the iteration task. Moreover, this validates that FWAdaBound takes the least computation cost among all the experimental algorithms. The reason is that FWAdaBound uses linear search instead of the high-order projection steps in Adam and AdaBound. Therefore, FWAdaBound can iterates much faster than Adam and AdaBound in each epoch.

Then, we execute the second experiment to verify the generalization ability of training accuracy. The results of the second experiment are shown in Figure 2. This figure shows that the training accuracy of FWAdaBound rises rapidly in the early stage of training and finally reaches the same height as AdaBound, which validates the generalization ability of training accuracy of FWAdaBound is better. Moreover, the figure also shows that the training accuracy of FWAdaBound is higher than that of Adam and AdaBound at each moment, which further indicates that the iteration cost of FWAdaBound is the lowest.

Finally, the last experiment verify the generalization ability of test accuracy of our proposed algorithm. And the results of this experiment are presented in Figure 3. Likewise, the test accuracy of FWAdaBound performs well in the early stage of iteration process and achieves the same performance as AdaBound in the final stage. Therefore, FWAdaBound also has a good generalization ability on test accuracy. In general, the generalization ability of FWAdaBound is the same as that of AdaBound but takes much less computation cost than AdaBound and Adam.

6.2.2. Language Modeling. In this group of experiments, we implement all the algorithms in 1-, 2-, and 3-layer LSTM models on the Penn Treebank dataset. The results of the experiments are shown in Figure 4 which presents the relationship between the perplexity and the running time of each algorithm. Note that the lower perplexity the better.

The experiment results of 1-layer LSTM show that FWAdaBound takes the least time to minimize the perplexity. On 1-layer LSTM model, FWAdaBound performs best in all algorithms, and Adam performs better than AdaBound.

In addition, in the experiments executed on 2-layer LSTM model, FWAdaBound has the quickest convergence rate among all the algorithms. Moreover, FWAdaBound takes 16.67% and 17.65% less running time than Adam and AdaBound, respectively. In this experiment, AdaBound performs better than Adam on the perplexity.

Finally, all the three algorithms are executed on the 3-layer LSTM model. The results show that FWAdaBound takes the least time to minimize the perplexity on Penn Treebank. In addition, FWAdaBound takes 11.83% and 14.05% less time than Adam and AdaBound, respectively. Moreover, AdaBound also performs better than Adam on the perplexity in this experiments.

7. Conclusion

In this paper, we proposed a Frank-Wolfe adaptive momentum online algorithm named FWAdaBound, which uses the Frank-Wolfe technique to avoid the projection operation. Moreover, our convergence analysis showed that the regret bound of FWAdaBound achieves to $O(T^{3/4})$, where T is a time horizon. In order to validate the performance of FWAdaBound in applications, we execute three groups of experiments for image classification and language modeling. The results show that FWAdaBound has good performance in the generalization ability of training and test accuracy.

Data Availability

The data that support the findings of this study are CIFAR-10 [21] and Penn Treebank datasets [22].

Conflicts of Interest

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

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grant Nos. 62002102, 72002133, and 62176113, in part by the Ministry of Education of China Science Foundation under Grant No. 19YJC630174, in part by the Key Technologies R & D Program of Henan Province under Grant No. 212102210083, and in part by the Luoyang Major Scientic and Technological Innovation Projects under Grant No. 2101017A.

References

[1] J. Li, R. Feng, W. Sun, Z. Liu, and Q. Li, "QoE-driven coupled uplink and downlink rate adaptation for 360-degree video live streaming," *IEEE Communications Letters*, vol. 24, no. 4, pp. 863–867, 2020.

[2] J. Li, C. Zhang, Z. Liu, R. Hong, and H. Hu, "Optimal volumetric video streaming with hybrid saliency based tiling," *IEEE Transactions on Multimedia*, 2022.

[3] K. Ota, M. Dao, V. Mezaris, and F. De Natale, "Deep learning for mobile multimedia: a survey," *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol. 13, no. 3s, article 34, pp. 1–22, 2017.

[4] Y. Zhou, M. Zhang, J. Zhu, R. Zheng, and Q. Wu, "A randomized block-coordinate Adam online learning optimization algorithm," *Neural Computing and Applications*, vol. 32, no. 16, pp. 12671–12684, 2020.

[5] Y. Zhou, X. Wang, M. Zhang, J. Zhu, R. Zheng, and Q. Wu, "MPCE: a maximum probability based cross entropy loss function for neural network classification," *IEEE Access*, vol. 7, pp. 146331–146341, 2019.

[6] Y. Lu, G. Lu, R. Lin, J. Li, and D. Zhang, "SRGC-Nets: sparse repeated group convolutional neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 8, pp. 2889–2902, 2020.

[7] P. Chriskos, C. Frantzidis, P. Gkivogkli, P. Bamidis, and C. Kourtidou-Papadeli, "Automatic sleep staging employing convolutional neural networks and cortical connectivity images," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 1, pp. 113–123, 2020.

[8] G. Wang, J. Qiao, J. Bi, Q. Jia, and M. Zhou, "An adaptive deep belief network with sparse restricted Boltzmann machines," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 10, pp. 4217–4228, 2020.

[9] J. Yu and X. Yan, "Whole process monitoring based on unstable neuron output information in hidden layers of deep belief network," *IEEE Transactions on Cybernetics*, vol. 50, no. 9, pp. 3998–4007, 2020.

[10] Y. Cao, R. Ji, L. Ji, G. Lei, H. Wang, and X. Shao, "l^2-MPTCP: a learning-driven latency-aware multipath transport scheme for industrial internet applications," *IEEE Transactions on Industrial Informatics*, 2022.

[11] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of Machine Learning Research*, vol. 12, pp. 2121–2159, 2011.

[12] T. Tieleman and G. Hinton, "RMSprop: Divide the gradient by a running average if its recent magnitude," in *Neural Networks for Machine Learning*, pp. 23–31, Coursera, California, CA, USA, 2012.

[13] M. Zeiler, "ADADELTA: an adaptive learning rate method," 2012, CoRR abs/1212.5701, http://arxiv.org/abs/1212.5701.

[14] D. Kingma and J. Ba, "Adam: a method for stochastic optimization. ICLR (Poster)," 2015, http://arxiv.org/abs/1412.6980.

[15] S. Reddi, S. Kale, and S. Kumar, "On the convergence of Adam and beyond. International conference on learning representations," 2018, https://openreview.net/forum?id=ryQu7f-RZ.

[16] L. Luo, Y. Xiong, and Y. Liu, "Adaptive gradient methods with dynamic bound of learning rate. International conference on learning representations," 2019, https://openreview.net/forum?id=Bkg3g2R9FX.

[17] H. Huang, C. Wang, and B. Dong, "Nostalgic Adam: weighing more of the past gradients when designing the adaptive learning rate," 2018, CoRR abs/1805.07557, http://arxiv.org/abs/1805.07557.

[18] N. Keskar and R. Socher, "Improving generalization performance by switching from Adam to SGD," 2017, CoRR abs/1712.07628, http://arxiv.org/abs/1712.07628.

[19] M. Jaggi, "Revisiting Frank-Wolfe: projection-free sparse convex optimization," *Proceedings of the 30th International Conference on Machine Learning*, vol. 28, no. 1, pp. 427–435, 2013.

[20] F. Huang, L. Tao, and S. Chen, "Accelerated stochastic gradient-free and projection-free methods," *Proceedings of the 37th International Conference on Machine Learning*, vol. 119, pp. 4519–4530, 2020.

[21] A. Krizhevsky, "Learning multiple layers of features from tiny images," 2009, Technical Report. Available: http://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf.

[22] M. Marcus, A. Bies, and B. Schasberger, "The Penn TreeBank: annotating predicate argument structure," in *Proc. the workshop on Human Language Technology*, Plainsboro, NJ, 1994.

Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 6245397, 10 pages
https://doi.org/10.1155/2022/6245397

Research Article

Issues of Clinical Identity Verification for Healthcare Applications over Mobile Terminal Platform

Sultan Ahmad[1], **Hikmat A. M. Abdeljaber**[2], **Jabeen Nazeer**[1], **Mohammed Yousuf Uddin**[3], **Velmurugan Lingamuthu**[4] and **Amandeep Kaur**[5]

[1]*Department of Computer Science, College of Computer Engineering and Sciences, Prince Sattam Bin Abdulaziz University, P.O. Box. 151, Alkharj 11942, Saudi Arabia*
[2]*Department of Computer Science, Faculty of Information Technology, Applied Science Private University, Amman, Jordan*
[3]*Department of Information System, College of Computer Engineering and Sciences, Prince Sattam Bin Abdulaziz University, P.O. Box. 151, Alkharj 11942, Saudi Arabia*
[4]*Department of Computer Science, School of Informatics and Electrical Engineering, Hachalu Hundesa Campus, Ambo University, Ethiopia*
[5]*University Centre for Research and Development, Department of Computer Science and Engineering, Chandigarh University, Gharuan, Mohali, India*

Correspondence should be addressed to Velmurugan Lingamuthu; velmurugan.lingamuthu@ambou.edu.et

Received 22 February 2022; Revised 15 March 2022; Accepted 19 March 2022; Published 19 April 2022

Academic Editor: Yuanlong Cao

According to recent research, attacks on USIM cards are on the rise. In a 5G setting, attackers can also employ counterfeit USIM cards to circumvent the identity authentication of specified standard applications and steal user information. Under the assumption that the USIM can be replicated, the identity authentication process of common mobile platform applications is investigated. The identity authentication tree is generated by examining the application behavior of user login, password reset, and sensitive operations. We tested 58 typical applications in 7 categories, including social communication and personal health. We found that 29 of them only needed the SMS verification code received by the USIM card to pass the authentication. In response to this problem, it is recommended to enable two-step verification and use USIM anti-counterfeiting methods to complete the verification.

1. Introduction

USIM (Global User Identity Module) [1] is widely used as an identification module for user identity in UMTS (Universal Mobile Telecommunication System) networks and is commonly used in various mobile devices to provide users with authentication services, short message services, etc. Compared with the SIM (Subscriber Identity Module), the USIM Card has been upgraded in application support and security. While the USIM card supports 3G/4G services, it is backward compatible with the 2G network supported by the SIM card. In the increasingly mature 5G network, the USIM card will play a more important role in entity authentication and information exchange.

The promotion of mobile devices and the development of mobile applications complement and promote each other. As of the third quarter of 2019, data from the National Bureau of Statistics show that the number of 4G mobile phone users in my country has increased by 10.1% year-on-year, and mobile Internet access traffic has increased by 34.9% year-on-year [2].

At the same time, 130,000 5G base stations have been built, and 5G communication technology and commercialization have ushered in rapid development. Globally, in

2018, users all over the world downloaded a total of 194 billion mobile applications (Apps), covering financial payment, social communication, travel, entertainment, audio and video applications, etc., of which online financial payment applications have developed rapidly. The number of user downloads has increased by 27.8% compared with 2017 [3, 4]. The subsequent data privacy risks continue to grow. Mobile devices equipped with USIM cards and the applications in them have become a part of people's lives. These applications often have operating rights for sensitive user information and authenticate the logged-in user's identity. When implementing identity authentication, SMS verification code has been widely used as a low-cost, easy-to-implement, and low-threshold verification method for users to learn. As a necessary carrier for receiving SMS verification codes, the USIM card will directly threaten the security of all USIM devices once it can be copied or forged, and it will inevitably pose a considerable threat to the identity authentication process of the App in the device and user privacy [5–7].

1.1. SIM/USIM Security Research Status. When a user uses a mobile device to communicate, the SIM/USIM card in the device needs to be authenticated and connected to the network first. Research shows that although the USIM card uses the MILENAGE algorithm to achieve two-way authentication [8, 9], and the SIM card uses the A3/A8 algorithm to achieve one-way authentication, they all face the possibility of being copied.

1.1.1. SIM/USIM Card Copy Attack. Miškovsky et al. [10] proposed a feasible differential power analysis (DPA) side-channel attack method based on the power signal difference in the USIM authentication process. In the MILENAGE algorithm, the differential power consumption analysis is performed by selecting the f5 function among them. The expected value of the round key used in the authentication parameter calculation and the OPc can be calculated with the help of the Pearson correlation coefficient [11]. Based on this, the attacker can complete the copy of the original USIM card within a few minutes only by using an oscilloscope, a smart card analyzer, and a personal computer and realize the authentication and normal communication with the AuC (Authentication Center) [12–14].

In addition, Saxena and Chaudhari [15] studied the A3/A8 algorithm based on COMP128, combined with the SRES response number of the A3 authentication algorithm to crack the pseudorandom number generator used by AuC, and can extract the customer authentication secret of AuC and SIM card.

A copy of the SIM card is now displayed. Tabassum [16] considered that the COMP128 authentication algorithm used by GSM (Global System for Mobile Communications) enables attackers to successfully extract the authentication key of SIM by brute force cracking and proposed the basic process and common methods for OTA copying of SIM cards. Xie et al. [17] proposed a side-channel attack method called partition attack, which can perform fast power consumption analysis on the divided lookup table structure in COMP128 to extract the authentication key. For CDMA technology, Chen et al. [18] analyzed the look-up table in the CAVE protocol and the cyclic shift operation in the AKA protocol and designed different power analysis methods, which cost a very short time on 8-bit microprocessors and SIM cards. Time can successfully extract the authentication key [19, 20].

In the 5G network environment, the AKA authentication protocol adopted by USIM is consistent with the main process and algorithm parameters of the 3G/4G AKA protocol in the NSA mode [21, 22]. Therefore, the security analysis methods and cracking methods of the USIM card in the 3G/4G environment are still effective in the 5G network [23, 24]. The above research shows that there are a large number of USIM cards that are easy to be copied in the domestic and foreign markets. Attackers only need to spend a few minutes of power consumption data collection time to achieve offline cracking and copying of the target user's USIM card. It only takes a few minutes to tens of minutes to complete the cracking and copying using a personal PC.

In order to deal with the risk of USIM cards being cracked by the abovementioned attack methods, many chip design companies at home and abroad have begun to study various chip protection methods and apply protection technologies to newly designed and taped-out USIM chips. However, compared with the repair of software vulnerabilities, the solution of chip security problems often requires the redesign and development of the chip and the tape out. This is undoubtedly a longer time. For the USIM chip, even if it has been produced with antiattack capability USIM cards, these chips must be widely used.

Operators are still required to carry out a large-scale recall or forced replacement of the issued USIM chips, which is obviously not feasible. This objectively causes a large number of USIM cards that can be copied to be used for a long time and widely in reality [25–27].

1.1.2. SMS Verification Code Application and Security. SMS verification codes have been widely used in authentication links such as logging in to applications on mobile platforms or resetting passwords. As a carrier and bridge for the USIM card to transmit information to the App, it represents the connection between a specific USIM card and the device holder. SMS verification code is essentially a time-based one-time password (TOTP, time-based one-time password) [28], and its architecture is shown in Figure 1.

The authentication and message transmission from MSC (Mobile Switching Center) to UE are often based on GSM or UMTS networks. In practice, attacks against SMS mostly occur during the authentication process between the device and the base station or after the user receives the SMS message. Yubo et al. [29] analyzed various SMS attack vectors and pointed out that installing malware on devices to steal data is a common attack method against SMS security. Kotkar and Game [30] implemented an attack method that allows the device to send SMS messages without user permission and prevents the device from receiving the messages. When the attacker uses the copied USIM card to

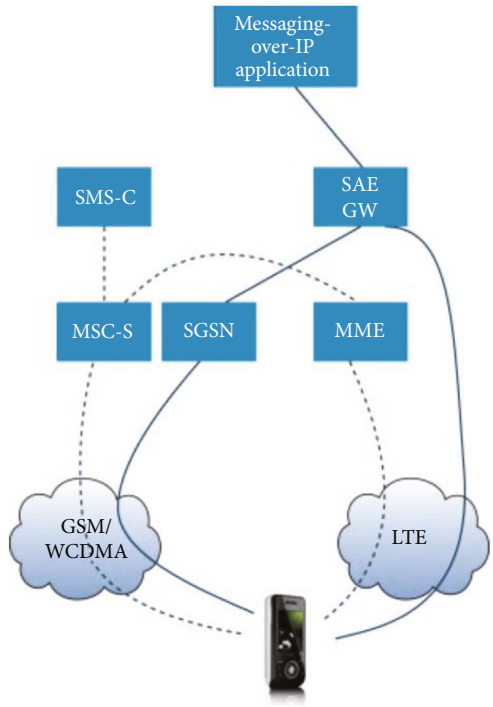


Figure 1: Overall structure of short message service (SMS) [7].

access the base station, he can receive the SMS verification code instead of the original user and complete the authentication process in the App.

1.2. The Main Work and Results of This Article. To avoid the security vulnerabilities or potential hazards of App applications created by the copied USIM card, the current practicable method is to update the App or security patches in a timely manner from the software level to compensate for the security risks caused by the usage of the copied USIM. In this context, this study analyzes and tests common apps in depth, identifying which apps have security issues when the USIM is duplicated and raising the alarm about the need for app updates and technology upgrades.

This article summarizes the general model of authentication for the general process of mobile application identity authentication and focuses on the analysis of possible security problems in the identity authentication strategy in the environment where the USIM card is copied. In an environment that simulates the USIM card being copied by an attacker, this paper studies the identity authentication link of 58 typical applications on the mobile terminal platform was tested, and the real machine test was carried out. The authentication process of application login, reset password, and sensitive operation was observed, and the application login data and jump process were analyzed and studied. The request-response data format and code execution process of these applications in the identity authentication process and finally recorded the use and performance of various security services (such as SMS verification codes) in the environment of copying the USIM card.

The test results found that 29 out of 58 applications have identity authentication services that can be directly bypassed in the environment where the USIM card is copied. Among them, 9 apps can be bypassed directly during password reset and login, 10 apps can be bypassed directly only during password reset, and the remaining 10 apps can be bypassed directly only during login. The test results in this article show that for applications that have identity authentication problems caused by USIM card copy attacks, mobile app developers, and security vendors should use at least two-step authentication and other software protection methods to avoid USIM card copy attacks on mobile applications and security risk.

2. Certification Process for Mobile Applications

Applications need to verify their identity before users perform functions. The identity authentication interfaces provided by various apps usually exist in user login, user reset the password, and performing sensitive operations. This section analyzes the general pattern of mobile application identity authentication and summarizes the users—identity verification tree.

2.1. Functional Scenarios of Identity Authentication. User login authentication: when a user accesses an application, the App needs to identify and authenticate the user's identity. User login usually requires a matching user name and password. The common user name types are usually user-defined strings, mailboxes, user mobile phone numbers, etc. After the password matches the user name, the current user will be allowed to log in. The general user login authentication process is shown in Figure 2. User reset password authentication: in practice, users forget their passwords from time to time, and user the application will also provide a password recovery function. After the user provides the correct user name, the authenticity of the username must be verified in conjunction with other information. The type of username determines the process of verifying identity. When using an email address or mobile phone number as a username, the application will send a verification email or SMS verification code. Only after the user receives the verification information and performs the corresponding operation will the password be allowed to reset, and some applications also adopt two-step verification and other means.

E-mail and mobile phones are an important part of daily life. Private mailbox letters and SMS verification codes are usually owned by individuals. Therefore, it is reasonable for application vendors to use them as necessary information for identity authentication, but some applications do not adopt additional verification methods to ensure current users. The correctness of the identity only guarantees the necessity of verification means. Once an external attacker manages to provide correct verification information, he can also reset the password and have the operation authority of the original user to achieve the purpose of the attack.

User authentication for sensitive operations: if a user performs certain sensitive operations after logging in, such as transferring money or viewing operation history, some applications will require the user to perform additional authentication, usually requiring the user to enter a PIN

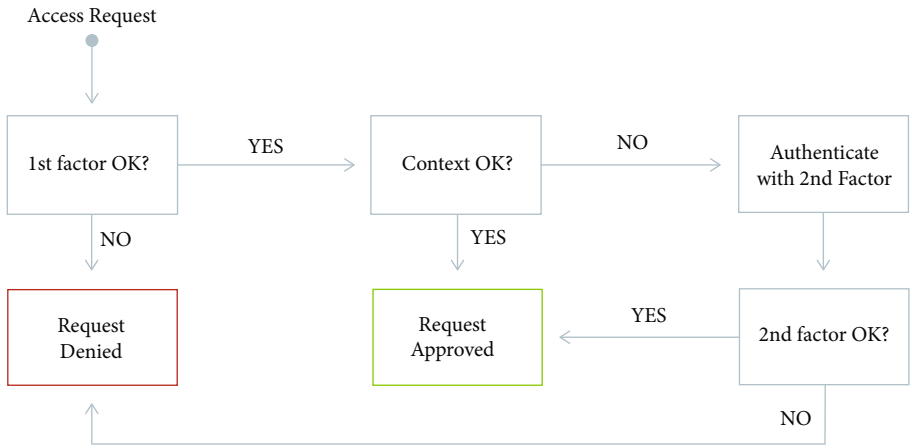


Figure 2: Procedure of authentication for App users logging in [12].

code, SMS verification code, or biometric verification. This measure can better protect the core business data. When personal financial apps involve banking services, it is often recommended that users turn on operation authentication to protect the safety of personal property.

2.2. The User Authentication Tree of the Application. According to the three common authentication processes in applications, this article can summarize the common user authentication trees in mobile applications. In the verification process, in order to achieve login applications, attackers can perform attacks on user login authentication and reset password authentication. In the process of user login authentication, potential logic vulnerabilities in the application can be used to initiate attacks; and in the process of resetting the password, there are many authentication methods involved, which also cause the attacker to have more attack originating points, such as SMS for USIM cards attack methods such as verification code security and email security. At the same time, because some applications have not developed a secondary verification process when resetting the password, the difficulty of the attack faced by the attacker is further reduced. Once the attacker successfully logs in to the application, and the application is not correct and sensitive operations are verified again, the attacker can obtain the response and use authority to steal user information.

3. Application Test of the Mobile Terminal Platform

This section adopts a field test method to examine the behavior of different applications in the certification process and introduces the standard test types and test techniques of mobile applications.

3.1. Mobile Application Testing and Analysis Technology. Common test types: various tests of mobile applications can help improve software quality to ensure long-term stable iterations of software versions. The main classification results of the test target and test method of the mobile application test method are shown in Figure 3.

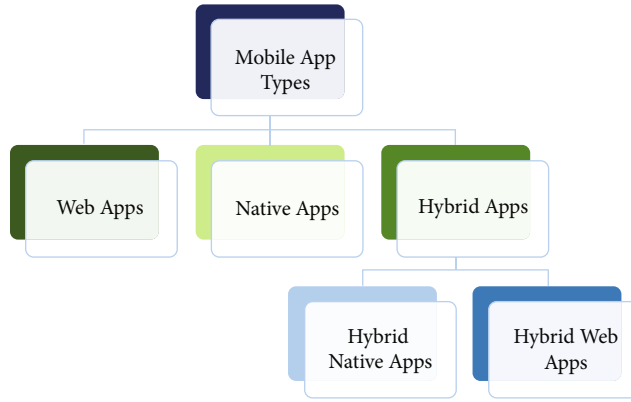


Figure 3: Classification of mobile applications [13].

Among them, functional testing examines the basic services, user interaction, and flexibility of the application. Safety testing, flow testing, power consumption testing, etc. have become necessary testing links in recent years [31]. Automated testing often uses a black box- (white box-) based automation framework to dynamically or statically analyze the product's modular units. In order to adapt to the rapid development and iteration of products, most manufacturers adopt automatic or semiautomatic testing methods.

Test analysis technology: the GUI automation framework API in the literature [32, 33] provides a common interface for the basic system functions of the mobile platform as the basis for other test functions. Testers write scripts through these APIs and use assert statements to test status information.

R&R-based test schemes such as "Reran" [34] and "Versatile" [35] may replace manual test scripts, and such methods can provide fine-grained capture and replay. The automatic input generation (AIG) technology [36] automates the generation process of test cases, which can improve code coverage, detect more errors, and reduce the scale of test programs. In addition, there are error reporting tools [37], equipment flow testing tools, etc.

Table 1: Tested mobile applications.

Category	Application composition
Social communication	WhatsApp, Messenger, Facebook, Instagram, Twitter, Skype, WeChat, Snapchat, Pinterest
Financial payment	Amazon, wish, eBay, Apple store, Walmart, Flipkart, cash, Paypal, Bank Internet Banking
Takeaway	Uber, Zomato, Parkmobile, Waze, UberEats, DoorDash, iFood, are you hungry
Health care	Keep, Nike Training Club, calm, pregnancy
File cloud disk	Dropbox, Google Drive, iCloud, Onedrive
Entertainment video	Youtube, TikTok, Netflix, Amazon Prime Video
Information retrieval	Google Chrome, search, Bing search

The researchers have analyzed the data interaction process and interaction strategy of several typical applications in the traditional test environment. Still, these applications have not been tested and researched in the environment where the USIM card is copied. This article makes up for this shortcoming and proposes to the behaviors and processes of App-like apps are tested to give out the security problems and deficiencies in the identity authentication of these apps and give solutions to them.

3.2. Identity Authentication Process Test for Typical Applications. In this section, in an environment where the attacker already has a copy of the USIM card, study the identity authentication of the test App in the USIM device, and analyze the data request and response results during the jump process of the identity authentication process by executing the identity authentication tree of different applications, observe and record the behavior of the App, and analyze the links that the attacker may bypass.

3.2.1. Test Conditions. Test object: the value of the information contained in the application is one of the main factors that affect the attacker's selection of attack targets. This test selects applications that are more likely to be targeted by the attacker. These applications usually occupy the mainstream market and can have a profound impact on the personal lives of a large number of users. These applications have a high user stickiness, are closely integrated with users' lives, and can access personal privacy and other data, which will be researched and targeted by attackers, and the potential security vulnerabilities of these applications will also lead to more serious data leakage incidents. Therefore, in order to improve the representativeness of the test results, combined with the abovementioned App analysis data, this article covers 7 types of applications in social communication, financial payment, travel delivery, health care, file cloud disk, entertainment video, and information retrieval. Launched the test, each type of application selected a total of 58 typical applications according to the download and usage rankings. The details are shown in Tables 1 and 2. The mobile big data service provider shows that applications such as short video, integrated e-commerce, and mobile payment have developed rapidly. These applications have a high user stickiness, are closely integrated with the user's life, can access personal privacy and other data, and will be subject to research and targeting by attackers, and these applications.

Table 2: Tested mobile platforms.

Index	Information
Model	iPhone XR
System	iOS 12.4.1
Operator	China Unicom
IMEI	357394092794037
ICCID	89860116208410304191
MEID	35739409279403

The potential security breaches will also lead to more serious data breaches. Therefore, in order to improve the representativeness of the test results, combined with the abovementioned App analysis data, this article covers 7 types of applications in social communication, financial payment, travel delivery, health care, file cloud disk, entertainment video, and information retrieval. A total of 58 typical applications are selected according to the ranking of downloads and usage. The details are shown in Tables 1 and 2.

The above 58 applications can access the user's communication content, property status, geographic location and travel trajectory, physical health status, private files, and retrieve information. This information is directly related to user privacy, and the leakage of this information will endanger user data security poses serious personal information security risks. Once these applications cannot guarantee the security of user identity verification, they will pose a greater potential threat to users' lives.

Test environment: the test model information used in this article is as follows.

Apple iOS and Google Android are the main types of mobile device systems. Except for the Apple authentication mechanism represented by iCloud, the remaining 57 apps have the same authentication process on iOS and Android. Therefore, the test results and conclusions of 58 applications of the equipment used in this test are consistent with the tests under different test systems result. Based on the above test conditions, the following prerequisites must be given before the test.

(1) The copied USIM card and the original USIM card are the same to the base station when sending and receiving messages

(2) The username of the target user and the mobile phone number of the USIM card are easy to obtain.

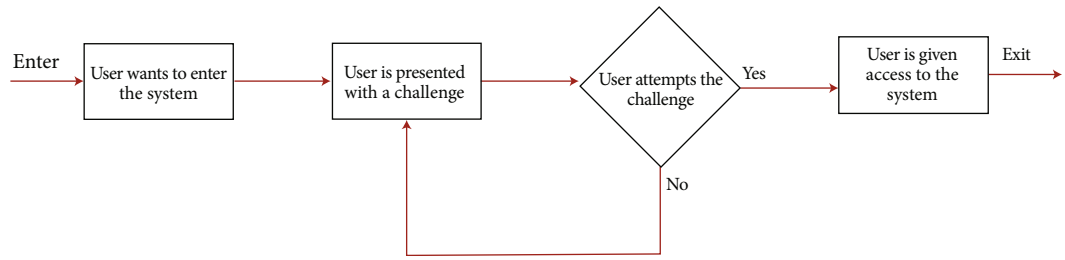


Figure 4: General procedure of authentication test [14].

According to the above test conditions, a general method of testing can be proposed

3.2.2. Test Methods. In this section, test the real device based on the verification tree of the mobile application. After downloading each application, use the target phone number or email address to register to complete the general registration process for new users. Then log out to simulate the behavior of an attacker, test the password reset function on the login page, pay attention to the response results of different steps and the links that require SMS verification codes, and record whether the application can be bypassed in password reset and other links. In the test process, use Stream to capture the Http/Https data request and response during the App login process, analyze the data packet fields and control methods, and combine the actual behavior of the App to give the test results. The basic test procedure is shown in Figure 4.

Other test requirements are as follows.

(1) For applications that can use mobile phone number or email to register and log in

In terms of usage, priority is given to using mobile phone to register and log in.

(2) For applications that use SSO services, the authentication logic is the same as that of the identity provider (IdP), and it is preferred not to use SSO services for testing

(3) For applications that do not enable secondary verification by default, keep the original settings for testing. If the application forces two-step verification to be turned on

Do not turn off this function during testing.

This article selects "Do you need SMS verification code," "Do you need secret information," "Do you need email verification," and "Do you verify the current device" as the test indicators in the password reset process. Combining the Http/Https data fields applied during the test and the control functions in the page code, if the user only needs the SMS verification code and does not need to verify the device environment to complete the password reset, it can be considered that the attacker is copying the USIM card environment. You can directly bypass the authentication link of the application, that is, the application is insecure.

3.3. Data Analysis in Identity Authentication of Typical Applications. During the testing process, this article captures and analyzes the request data and response results of the application. This section takes two typical applications of WeChat and Alipay as examples to analyze the format and jump flow of the request and response data in the process of resetting the password. And combined with the application behavior, give the test results of the two in the copied USIM environment.

3.3.1. Data Capture and Analysis in WeChat Login. According to the requirements in Section 3.2.2, this section needs to test the authentication process of WeChat to retrieve the password. Since WeChat can directly use the mobile phone number and SMS verification code to complete the login, when the user selects the "Retrieve Password" function, WeChat will make users taste.

Try to log in directly with your mobile phone number. In this process, Stream1.0.4 is used to capture and analyze Http network traffic. WeChat login process.

After analyzing the server response data, it can be seen that the above three request data correspond to the three stages of "request for password retrieval," "request for mobile verification code login," and "request for login application" when the client retrieves the password. The key fields and the code execution process of the response page expand the description of the identity authentication process during the login phase of WeChat.

Request to retrieve the password: when the user requests to retrieve the password on the WeChat login page, the client will initiate a request to support. After the verification is passed, the server responds, and the client jumps to the prompt message page for retrieving the password. At this time, WeChat will prompt the user to log in with a mobile phone. The response data includes the control code of the page, select the function button of "Can receive SMS," the ican function in the corresponding code will realize the jump to the next page. After confirming on the next page, the go function constructs a request for the mobile phone verification code to log in to the application.

Request mobile phone verification code login: when the user can log in with the SMS verification code, the user will be redirected to the login page using the SMS verification code according to the prompt. Among them, the go function sets the p1 and p2 fields to 1 and assigns rid to the p10 field. The above fields are spliced and used as report data, which is passed as part of the login request in the next stage to the

server. At this stage, the client will construct an ap_msg field that contains information about the current device's network environment and system parameters and use the GET method to request the server to send the verification code to the login page.

Request to log in to the application: when the user logs in using the SMS verification code, the 6-digit SMS verification code is required. After the client is authenticated, the reportFunc function uses the characteristics of the Image object of Javascript and realizes the static of the server resources by changing the source link attribute of the object access. Finally, the report data field in the message can tell the identity of the server user and the method used to log in to the application and use the GET method to send it to the server to complete the login.

In summary, in the process of retrieving the password and logging into WeChat, the client uses scan and go functions to complete the application identity authentication process based on the user's existing identity credentials. On the client-side, when the attacker chooses to log in with the SMS verification code during the password reset process. When used, you can copy the USIM card to directly obtain the SMS verification code to complete the login.

3.3.2. Data Analysis in Password Reset. Use the same method as in Section 3.3.1 to analyze password reset process analysis. When logging in to your account, this article chooses to reset the password instead of the SMS verification code to log in. Use Stream to capture the Https data in password.

The above request data packets correspond to the client's request to retrieve the password, send the SMS verification code, and send the reset password, respectively. When resetting the password, the client will repeatedly send to the host of http://abc.com/ the data including the phone model, network environment, operator type, and other device environment parameters, and the data will be returned by the server.

The status code determines whether the current device can continue the next operation. Due to the large amount of Https traffic data in this link, only the three request response processes will be explained below.

4. Identity Authentication Test Results and Analysis of Typical Applications

According to the test requirements and test methods in Section 3, through the process observation and recording of the identity authentication functions such as login and password retrieval of the test application, this section presents the test results and analysis.

4.1. Overview of Test Results. According to records, when resetting the password, 19 of the 58 apps used in the test can be directly used to reset the password through the SMS verification code to complete the login. The remaining 39 apps are attacked due to additional requirements such as confidential information, email verification, and device verification.

Apps cannot be bypassed because the app only provides email verification but not SMS verification. Except for Apple's App store and iCloud, the verification methods for other test applications are SMS verification or email verification. Among the 21 apps that performed secondary verification, 4 adopted additional confidential information, 4 adopted additional email verification, and 13 apps tested the current device environment to remind or warn the original logged-in user of the current attacker. Behavior to prevent attackers from bypassing directly. Some of these 21 applications also use multiple verification methods to ensure product safety.

The test also found that when the attacker can obtain the SMS verification code sent to the user's device, 9 apps can be bypassed during password reset and normal login, and 10 apps are easily bypassed only during password reset. However, there are 10 other applications that can be bypassed only during normal login. Among them, the application names included in each indicator are shown in Table 3.

When at least one verification method is additionally adopted, it is difficult for an attacker to steal user information. However, when logging in normally, the attacker can use the user's mobile phone number to log in, which makes each application default to the original legitimate user who is currently logged in remotely, so no additional verification is performed.

4.2. Application Classification and Index Test Results. For each test indicator, 38 apps provide an interface for SMS verification to reset passwords, and only 4 apps require users to provide confidential information, such as historical orders and purchased product names. For email verification, 4 apps require SMS verification. In the case of, additional email verification is still required, while 22 models only require email verification, and 16 apps will perform environmental testing of newly logged-in devices, etc.

Seven types of applications tested, the proportion of financial applications that are directly bypassed is the least, only 20%. Among the 20 apps in social, health, and entertainment, 11 can be bypassed. In the global mobile application market, applications that directly involve the safety of users' personal property have relatively safe protection measures, while applications that indirectly involve user privacy, such as providing entertainment and personal health information, still have relatively simple identity authentication strategies and are vulnerable to attackers.

4.3. Analysis and Research of Test Results. Combining the test results, this section analyzes the current status and characteristics of the mobile application's identity authentication process from the perspective of the characteristics of the application process, the relationship between the application type and the authentication process.

4.3.1. Differences in SMS Verification Services at Home and Abroad. During the test, it was noticed that most of the apps in India have implemented the service entrance, while many foreign apps only use email addresses and user names for authentication. Further analysis found that all 17 domestic applications provide SMS verification and reset password

Table 3: Comparison between domestic and foreign Apps.

	SMS verification	Reset password services	Directly bypassable
Domestic apps	17	17	6
Foreign apps	41	21	13

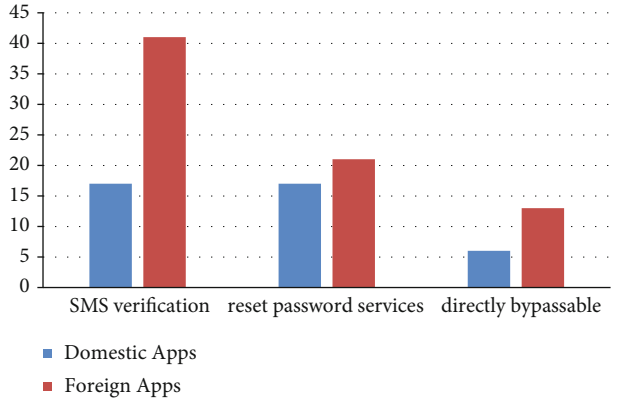


Figure 5: Comparison between domestic and foreign Apps.

services, and 6 of them are deemed to be directly bypassable, while 21 of foreign applications provide SMS verification, and 13 of them can be bypassed, such as shown in Figure 5.

The reason why mobile Internet companies and relevant departments implement SMS verification code services on a large scale is not only because of the low cost and difficulty of operation of SMS verification codes but also the effective supervision of mobile applications. The test also found that foreign apps do not require mobile phone number binding, and even a small number of apps do not have a mobile phone number registration entry.

4.3.2. Lack of Confidential Information. For the test results of each indicator, only 6.9% of applications use secret information as the second step of verification. For different types of confidential information, too high uniqueness may increase the user's operation difficulty, while low uniqueness will appear very fragile in the face of various social engineering methods, as shown in Figure 6.

Secret information required by the application is the last four digits of a random merchant order number in the user's previous orders. This operation is difficult for users who do not frequently use mobile smartphones. The application requires any of the historical orders to be filled in the name of a consignee, and this information is easily leaked by various phishing and retrieval methods. Therefore, the design of confidential information with low user operation difficulty but high uniqueness should become a problem that needs to be considered in the application of the App user identity verification system in the future.

4.3.3. Value Difference of Application Information Type. For different types of applications, having a low bypass ratio only indicates that it has a better security performance in the

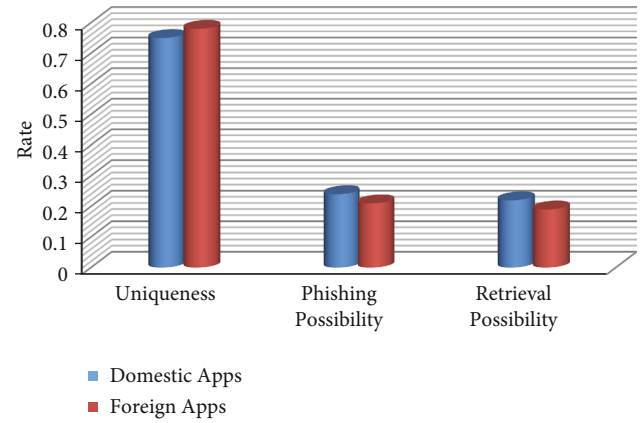


Figure 6: Confidentiality level between domestic and foreign Apps.

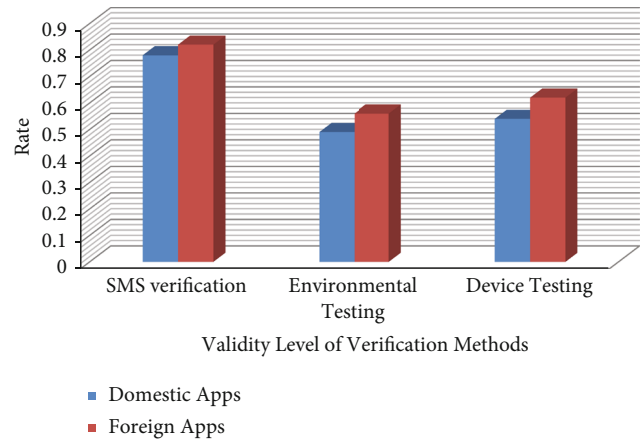


Figure 7: Validity level of verification methods.

Table 4: Verification label validity.

	Validity level of verification methods		
	SMS verification	Environmental testing	Device testing
Domestic apps	0.78	0.49	0.54
Foreign apps	0.82	0.56	0.62

environment of copying the USIM card, but it does not indicate that it has a strong degree of protection in protecting user information. More retail applications and a small number of payment applications do not require additional identity binding when registering, and the secondary verification function is not mandatory. It can be considered that in the information value evaluation system of users and mobile Internet companies, compared with personal online assets (such as all property, cloud files, etc.), the sensitivity of personal interests, health conditions, and social content is not high.

Making entertainment, health care, and social networking applications have a single verification method, resulting in a higher percentage of bypassable applications for these three types of applications.

4.3.4. Improvement of Equipment Verification Methods. The test found that when resetting the password, 27.6% of the applications will detect the user's device model and environment, such as capturing the current network IP address, geographic location, and machine hardware parameters, and log in frequently with the original device. Information matching and peer recognition of the user who tried to log in before or remind the original device user.

At present, there are three main methods for the verification of new devices: SMS verification, device testing, and environmental testing as shown in Figure 7 and Table 4. The device detection link is one of the two-step verification methods adopted by apps such as Google when they detect a new device login. When an attacker tries to reset the password on a device, the application that the original device logs in will prompt the user whether to allow the operation to occur. Only after the authentication is completed in the device, the user account can request other devices to reset the password. For example, when an attacker tries to reset the password of a target Apple ID on a certain device, Apple requires the ID to log in to other devices for first verification. This can effectively prevent attackers from using the copied USIM card to log in to the application.

5. Conclusion

This paper focuses on the problem of mobile application identity authentication. It first explains the authentication process of mobile applications and describes the general verification tree of mobile applications to identify problematic linkages in the identity authentication strategy; then, when various mobile applications authenticate users, log in to the application. The authentication methods used in resetting passwords and completing sensitive actions may be the same, whereas SMS verification codes and email verification are commonly employed in the process of resetting passwords and login authentication. The USIM card copy attack makes it possible for the attacker to obtain the SMS verification code sent to the original user device, so that the password reset and application login in the verification tree can be used as the starting point of the attack, and user information can be stolen after bypassing the authentication login application.

This article uses real machine testing to simulate the security flaws of password reset and login authentication of various applications after the attacker successfully implements the USIM card copy attack. After analyzing the Https traffic data in the identity authentication process and combining the application behaviors, it is found that a total of 29 of 58 applications face security risks that can be directly bypassed by copying the USIM card. Among them, 9 applications can be used for password reset and login. It is bypassed directly. There are 10 apps that can be bypassed only when the password is reset, and the remaining 10 apps can only be bypassed directly when logging in.

Among them, social communication and entertainment applications are most easily bypassed. Mobile application developers and security personnel should complete and improve the authentication logic of current products, adopt two-step authentication or two-factor authentication to prevent USIM card copy attacks, and implement effective deployment of user data security protection methods.

Data Availability

The data used to support the findings of this study are available from the author upon request (s.alisher@psau.edu.sa).

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Authors' Contributions

Sultan Ahmad wrote the paper, Hikmat A. M. Abdeljaber validated the paper, Jabeen Nazeer designed the methodology, Mohammed Yousuf Uddin proofread the paper, Velmurugan Lingamuthu validated the software, and Amandeep Kaur proposed the method.

Acknowledgments

The authors are grateful to the Applied Science Private University, Amman-Jordan, for the full financial support granted to cover the publication fee of this research article.

References

[1] W. F. El-Sadek and M. N. Mikhail, "Universal mobility with global identity (UMGI) architecture," *International Conference on Wireless Networks and Information Systems*, vol. 2009, pp. 389–394, 2009.

[2] B. Mathur and S. M. Satapathy, "An analytical comparison of mobile application development using agile methodologies," in *2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI)*, pp. 1147–1152, Tirunelveli, India, 2019.

[3] S. Khan, Z. Jiangbin, and A. Wahab, "Design and development of android performance testing tool," in *IEEE Conference on Big Data and Analytics (ICBDA)*, pp. 57–60, Kota Kinabalu, Malaysia, 2020.

[4] V. P. La Manna and F. Pasveer, "Towards a framework for proximity-based hybrid mobile applications," in *Proceedings of the 5th International Conference on Mobile Software Engineering and Systems*, pp. 176–179, 2018.

[5] G. Dhiman, J. Rashid, J. Kim, S. Juneja, W. Viriyasitavat, and K. Gulati, "Privacy for healthcare data using the byzantine consensus method," *IETE Journal of Research*, pp. 1–12, 2022.

[6] S. Kanwal, J. Rashid, J. Kim, S. Juneja, G. Dhiman, and A. Hussain, "Mitigating the coexistence technique in wireless body area networks by using superframe interleaving," *IETE Journal of Research*, pp. 1–15, 2022.

[7] N. Singh, E. H. Houssein, S. B. Singh, and G. Dhiman, "HSSAHHO: a novel hybrid Salp swarm-Harris hawks optimization algorithm for complex engineering problems," *Journal of Ambient Intelligence and Humanized Computing*, pp. 1–37, 2022.

[8] G. Dhiman, A. K. Nagar, S. Vimal, and S. Rho, "Cybertwin-Driven 6G for Internet of Everything (IoE): Architectures," *IEEE Transactions on Industrial Informatics*, 2022.

[9] G. Dhiman, S. Juneja, W. Viriyasitavat et al., "A novel machine-learning-based hybrid CNN model for tumor identification in medical image processing," *Sustainability*, vol. 14, no. 3, p. 1447, 2022.

[10] V. Miškovsky, H. Kubátová, and M. Novotný, "Speeding up differential power analysis using integrated power traces," in *2018 7th Mediterranean Conference on Embedded Computing (MECO)*, pp. 1–5, Budva, Montenegro, 2018.

[11] J. Liu, Y. Zhang, and Q. Zhao, "Video stabilization algorithm based on Pearson correlation coefficient," *International Conference on Advanced Mechatronic Systems (ICAMechS)*, vol. 2019, pp. 289–293, 2019.

[12] K. Kour, D. Gupta, K. Gupta et al., "Smart-hydroponic-based framework for saffron cultivation: a precision smart agriculture perspective," *Sustainability*, vol. 14, no. 3, p. 1120, 2022.

[13] G. Dhiman, A. Vignesh Kumar, R. Nirmalan et al., "Multimodal active learning with deep reinforcement learning for target feature extraction in multi-media image processing applications," *Multimedia Tools and Applications*, pp. 1–25, 2022.

[14] K. Prasanna, K. Ramana, G. Dhiman, S. Kautish, and V. D. Chakravarthy, "PoC design: a methodology for proof-of-concept (PoC) development on internet of things connected dynamic environments," *Security and Communication Networks*, vol. 2021, Article ID 7185827, 2021.

[15] N. Saxena and N. S. Chaudhari, "Secure algorithms for SAKA protocol in the GSM network," in *2017 10th IFIP Wireless and Mobile Networking Conference (WMNC)*, pp. 1–8, Valencia, Spain, 2017.

[16] K. Tabassum, "An efficient authentication technique for security management against cloning mobile phones," in *2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS)*, pp. 125–128, Chennai, India, 2017.

[17] H. Xie, K. Lv, and C. Hu, "A partition matching method for optimal attack path analysis," in *2018 IEEE Intl Conf on parallel & distributed processing with applications, Ubiquitous Computing & Communications, Big Data & Cloud Computing, Social Computing & Networking, Sustainable Computing & Communications (ISPA/IUCC/BDCloud/SocialCom/SustainCom)*, pp. 120–126, 2018.

[18] Z. Chen, L. Zhang, W. Wang, and Z. Wu, "A pre-coded multicarrier M-ary chaotic vector cyclic shift keying transceiver for reliable communications," *IEEE Transactions on Wireless Communications*, vol. 21, no. 2, pp. 1007–1021.

[19] S. Kranthi Kumar, K. Ramana, G. Dhiman, S. Singh, and B. Yoon, "A novel blockchain and bi-linear polynomial-based QCP-ABE framework for privacy and security over the complex cloud data," *Sensors*, vol. 21, no. 21, p. 7300, 2021.

[20] G. Dhiman and R. Sharma, "SHANN: an IoT and machine-learning-assisted edge cross-layered routing protocol using spotted hyena optimizer," *Complex & Intelligent Systems*, pp. 1–9, 2021.

[21] M. Ouaissa, M. Houmer, and M. Ouaissa, "An enhanced authentication protocol based group for vehicular communications over 5G networks," in *2020 3rd International Conference on Advanced Communication Technologies and Networking (CommNet),*, pp. 1–8, Marrakech, Morocco, 2020.

[22] K. Han, M. Ma, X. Li, Z. Feng, and J. Hao, "An efficient handover authentication mechanism for 5G wireless network," *IEEE Wireless Communications and Networking Conference (WCNC)*, vol. 2019, pp. 1–8, 2019.

[23] X. G. Huang, L. Shen, and Y. H. Feng, "A user authentication scheme based on fingerprint and USIM card," in *2008 International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, pp. 1261–1264, Harbin, China, 2008.

[24] J. Cao, Z. Yan, R. Ma, Y. Zhang, Y. Fu, and H. Li, "LSAA: a lightweight and secure access authentication scheme for both UE and mMTC devices in 5G networks," *IEEE Internet of Things Journal*, vol. 7, no. 6, pp. 5329–5344, 2020.

[25] V. K. Gupta, S. K. Shukla, and R. S. Rawat, "Crime tracking system and people's safety in India using machine learning approaches," *International Journal of Modern Research*, vol. 2, no. 1, pp. 1–7, 2022.

[26] P. K. Vaishnav, S. Sharma, and P. Sharma, "Analytical review analysis for screening COVID-19 disease," *International Journal of Modern Research*, vol. 1, no. 1, pp. 22–29, 2021.

[27] R. Kumar and G. Dhiman, "A comparative study of fuzzy optimization through fuzzy number," *International Journal of Modern Research*, vol. 1, no. 1, pp. 1–14, 2021.

[28] B. Reaves, N. Scaife, D. Tian, L. Blue, P. Traynor, and K. R. B. Butler, "Sending out an SMS: characterizing the security of the SMS ecosystem with public gateways," *IEEE Symposium on Security and Privacy (SP)*, vol. 2016, pp. 339–356, 2016.

[29] S. Yubo, Z. Zhiwei, and X. Yunfeng, "Using short message service (SMS) to deploy android exploits," in *International Conference on Cyberspace Technology (CCT)*, pp. 1–5, Beijing, China, 2014.

[30] C. Kotkar and P. Game, "Prevention mechanism for prohibiting SMS malware attack on android smartphone," *Annual IEEE India Conference (INDICON)*, vol. 2015, pp. 1–5, 2015.

[31] S. Rinaldi, M. Pasetti, A. Flammini, P. Ferrari, E. Sisinni, and F. Simoncini, "A testing framework for the monitoring and performance analysis of distributed energy systems," *IEEE Transactions on Instrumentation and Measurement*, vol. 68, no. 10, pp. 3831–3840, 2019.

[32] W. Squires and P. Centonze, "Cross-platform access-rights analysis of mobile applications," *IEEE/ACM International Conference on Mobile Software Engineering and Systems (MOBILESoft)*, vol. 2016, pp. 295-296, 2016.

[33] W. Huang, Z. Chen, W. Dong, H. Li, B. Cao, and J. Cao, "Mobile internet big data platform in China Unicom," *Tsinghua Science and Technology*, vol. 19, no. 1, pp. 95–101, 2014.

[34] L. Gomez, I. Neamtiu, T. Azim, and T. Millstein, "RERAN: timing- and touch-sensitive record and replay for android," in *2013 35th International Conference on Software Engineering (ICSE)*, pp. 72–81, San Francisco, CA, USA, 2013.

[35] S. Peng, Z. Peng, Y. Ren, and F. Chen, "Fast intra-frame coding algorithm for versatile video coding based on texture feature," in *2019 IEEE International Conference on Real-time Computing and Robotics (RCAR)*, pp. 65–68, Irkutsk, Russia, 2019.

[36] Q. Liu and Q. Liu, "Research on automatic generation control system of photovoltaic power station based on adaptive PID control algorithm," in *2020 IEEE 3rd International Conference on Information Systems and Computer Aided Education (ICISCAE),*, pp. 231–236, Dalian, China, 2020.

[37] M. Su, Y. Wang, N. Yin, and H. Liu, "The effects of position errors compensation on the other geometric errors in the CNC machine tools," in *2019 6th International Conference on Information Science and Control Engineering (ICISCE)*, pp. 939–947, Shanghai, China, 2019.