

# Artificial Intelligence Techniques for Securing Smart Devices

Lead Guest Editor: Muhammad Ahmad

Guest Editors: Adil Mehmood Khan, Usman Habib, and Manuel Mazzara





---

# **Artificial Intelligence Techniques for Securing Smart Devices**

Security and Communication Networks

---

## **Artificial Intelligence Techniques for Securing Smart Devices**

Lead Guest Editor: Muhammad Ahmad

Guest Editors: Adil Mehmood Khan, Usman Habib,  
and Manuel Mazzara



---





Copyright © 2022 Hindawi Limited. All rights reserved.

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

# Chief Editor

Roberto Di Pietro, Saudi Arabia

## Associate Editors

Jiankun Hu , Australia  
Emanuele Maiorana , Italy  
David Megias , Spain  
Zheng Yan , China

## Academic Editors

Saed Saleh Al Rabae , United Arab Emirates  
Shadab Alam, Saudi Arabia  
Goutham Reddy Alavalapati , USA  
Jehad Ali , Republic of Korea  
Jehad Ali, Saint Vincent and the Grenadines  
Benjamin Aziz , United Kingdom  
Taimur Bakhshi , United Kingdom  
Spiridon Bakiras , Qatar  
Musa Balta, Turkey  
Jin Wook Byun , Republic of Korea  
Bruno Carpentieri , Italy  
Luigi Catuogno , Italy  
Ricardo Chaves , Portugal  
Chien-Ming Chen , China  
Tom Chen , United Kingdom  
Stelvio Cimato , Italy  
Vincenzo Conti , Italy  
Luigi Coppolino , Italy  
Salvatore D'Antonio , Italy  
Juhriyansyah Dalle, Indonesia  
Alfredo De Santis, Italy  
Angel M. Del Rey , Spain  
Roberto Di Pietro , France  
Wenxiu Ding , China  
Nicola Dragoni , Denmark  
Wei Feng , China  
Carmen Fernandez-Gago, Spain  
AnMin Fu , China  
Clemente Galdi , Italy  
Dimitrios Geneiatakis , Italy  
Muhammad A. Gondal , Oman  
Francesco Gringoli , Italy  
Biao Han , China  
Jinguang Han , China  
Khizar Hayat, Oman  
Azeem Irshad, Pakistan




M.A. Jabbar , India  
Minho Jo , Republic of Korea  
Arijit Karati , Taiwan  
ASM Kayes , Australia  
Farrukh Aslam Khan , Saudi Arabia  
Fazlullah Khan , Pakistan  
Kiseon Kim , Republic of Korea  
Mehmet Zeki Konyar, Turkey  
Sanjeev Kumar, USA  
Hyun Kwon, Republic of Korea  
Maryline Laurent , France  
Jegatha Deborah Lazarus , India  
Huaizhi Li , USA  
Jiguo Li , China  
Xueqin Liang, Finland  
Zhe Liu, Canada  
Guangchi Liu , USA  
Flavio Lombardi , Italy  
Yang Lu, China  
Vincente Martin, Spain  
Weizhi Meng , Denmark  
Andrea Michienzi , Italy  
Laura Mongioi , Italy  
Raul Monroy , Mexico  
Naghme Moradpoor , United Kingdom  
Leonardo Mostarda , Italy  
Mohamed Nassar , Lebanon  
Qiang Ni, United Kingdom  
Mahmood Niazi , Saudi Arabia  
Vincent O. Nyangaresi, Kenya  
Lu Ou , China  
Hyun-A Park, Republic of Korea  
A. Peinado , Spain  
Gerardo Pelosi , Italy  
Gregorio Martinez Perez , Spain  
Pedro Peris-Lopez , Spain  
Carla Ràfols, Germany  
Francesco Regazzoni, Switzerland  
Abdalhossein Rezai , Iran  
Helena Rifà-Pous , Spain  
Arun Kumar Sangaiah, India  
Nadeem Sarwar, Pakistan  
Neetesh Saxena, United Kingdom  
Savio Sciancalepore , The Netherlands

De Rosal Ignatius Moses Setiadi ,  
Indonesia  
Wenbo Shi, China  
Ghanshyam Singh , South Africa  
Vasco Soares, Portugal  
Salvatore Sorce , Italy  
Abdulhamit Subasi, Saudi Arabia  
Zhiyuan Tan , United Kingdom  
Keke Tang , China  
Je Sen Teh , Australia  
Bohui Wang, China  
Guojun Wang, China  
Jinwei Wang , China  
Qichun Wang , China  
Hu Xiong , China  
Chang Xu , China  
Xuehu Yan , China  
Anjia Yang , China  
Jiachen Yang , China  
Yu Yao , China  
Yinghui Ye, China  
Kuo-Hui Yeh , Taiwan  
Yong Yu , China  
Xiaohui Yuan , USA  
Sherali Zeadally, USA  
Leo Y. Zhang, Australia  
Tao Zhang, China  
Youwen Zhu , China  
Zhengyu Zhu , China

## Contents

---

### **Android Malware Detection Technology Based on Lightweight Convolutional Neural Networks**

Genchao Ye , Jian Zhang , Huanzhou Li, Zhangguo Tang, and Tianzi Lv 





Research Article (12 pages), Article ID 8893764, Volume 2022 (2022)

### **5G and Blockchain Enabled Lightweight Solutions for Containing COVID-19**

Mohsin Kamal , Abdulah Jeza Aljohani , Eisa Alanazi, and Fahad R. Albogamy

Research Article (12 pages), Article ID 3370408, Volume 2022 (2022)

### **Image Splicing-Based Forgery Detection Using Discrete Wavelet Transform and Edge Weighted Local Binary Patterns**

Muhammad Hameed Siddiqi , Khurshed Asghar, Umar Draz , Amjad Ali , Madallah Alruwaili ,

Yousef Alhwaiti , Saad Alanazi , and M. M. Kamruzzaman 




Research Article (10 pages), Article ID 4270776, Volume 2021 (2021)

### **StFuzzer: Contribution-Aware Coverage-Guided Fuzzing for Smart Devices**

Jiageng Yang, Xinguo Zhang, Hui Lu , Muhammad Shafiq, and Zhihong Tian 



Research Article (15 pages), Article ID 1987844, Volume 2021 (2021)

### **Security-Oriented Indoor Robots Tracking: An Object Recognition Viewpoint**

Yaoqi Yang , Xianglin Wei , Renhui Xu, Laixian Peng , Yunliang Liao, and Lin Ge

Research Article (9 pages), Article ID 7456552, Volume 2021 (2021)

### **Acquiring Data Traffic for Sustainable IoT and Smart Devices Using Machine Learning Algorithm**

Yi Huang, Shah Nazir , Xinqiang Ma , Shiming Kong, and Youyuan Liu

Research Article (11 pages), Article ID 1852466, Volume 2021 (2021)

## Research Article

# Android Malware Detection Technology Based on Lightweight Convolutional Neural Networks

Genchao Ye <sup>1,2</sup>, Jian Zhang <sup>1,2</sup>, Huanzhou Li,<sup>1,2</sup> Zhangguo Tang,<sup>1,2</sup> and Tianzi Lv <sup>1,2</sup>

<sup>1</sup>School of Physics and Electronic Engineering, Sichuan Normal University, Chengdu 610101, China

<sup>2</sup>Institute of Network and Communication Technology, Sichuan Normal University, Chengdu 610101, China

Correspondence should be addressed to Jian Zhang; zhangjian@sicnu.edu.cn

Received 22 May 2021; Revised 9 July 2021; Accepted 23 February 2022; Published 16 March 2022

Academic Editor: Usman Habib

Copyright © 2022 Genchao Ye et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

With the rapid development of Android, a major mobile Internet platform, Android malware attacks have become the number one threat to mobile Internet security. Traditional malware detection methods have low precision and greater time complexity. At present, image detection methods based on deep learning are used in malware detection. However, most of these methods are based on the largescale convolutional neural network model (such as VGG16). The computation and weight files of these models are very large, so they are not suitable for mobile Internet platforms with limited computation. A novel detection method based on a lightweight convolutional neural network is presented in this study. It transforms Android malware classes.dex, Android-manifest.xml, and resource.arsc into RGB images and uses the lightweight convolutional neural network to extract the features of RGB images automatically. The experimental results of this study indicate that the method performs well in terms of precision and speed of detection.

## 1. Introduction

Android is one of the most common smartphone operating systems. According to the statistics of the global smartphone market system occupancy in the second quarter of 2018 by Kantar, an international data research organization, 82% of the market share was occupied by Android, and the share is still increasing. The rapid popularization of Android has not only brought great convenience to users but also attracted the attention of network hackers. Due to the openness of Android, Android devices are susceptible to malware infection. AV-TEST Security Report 2019/2020 [1] shows that 6,201,358 new Android malware samples were added in 2017, and the number of new malware samples has increased from 2019. In addition, Android won the first place in 2019 with 417 known security vulnerabilities in the ranking of operating systems and programs with the number of security vulnerabilities and was listed in the common vulnerabilities and exposure (CVE) database. Among all Android malware in 2019, Trojan accounted for more than 90%, followed by malicious adware and

ransomware, which are also the main ways for criminals to illegally obtain benefits.

Obviously, it is of extreme urgency to detect and prevent malware in order to resolve serious security problems caused by Android malware. In the past, static analysis technology and dynamic analysis technology [2] were the main methods of Android malware detection. Static analysis [3] refers to the extraction and analysis of the features of the software without running. Lei [4] extracted the API call sequence of the Android software through reverse engineering and normalization and then detected it through a probability discriminant learning model based on regular logistic regression. MaMaDroid [5] constructed the behavior model in the form of Markov chain for the API call sequence and classified malware. Tian [6] solved the problem of repackaged malware detection by analyzing the dependence between classes and classes, methods, and methods according to the heterogeneity of Android malware code. Kong [7] extracted structural information on the malware program as an attribute function call graph and then used an integrated classifier for automatic malware classification. However, it is



very difficult to the reverse engineering of the Android software for static analysis, and it is easy to be interfered by code obfuscation technology or encryption technology, resulting in inaccurate detection results. Dynamic analysis technology [8] refers to executing malware in a secure virtual environment and monitoring and extracting malware behavior information dynamically. The MADAM system [9] could judge whether it is malicious according to the seven abnormal behaviors of the software when it is running. Yuan [10] performed a dynamic taint analysis through system hooks and monitored various application operations. Each behavior feature was one-hot encoded and transmitted to the Deep Belief Networks (DBN) as trained data. DroidTrace [11] was a dynamic analysis system with forward execution capability based on the Ptrace which is a dynamic analysis tool. It could judge whether it is malware based on the system dynamic load when the software is running. Somarriba [12] provided a framework of monitoring and visualized abnormal function calls of Android applications. Dynamic analysis technology has a high accuracy rate, but repeated code execution is time-consuming and resource intensive.

In recent years, visualization technology has been introduced into malware detection. Malware visualization technology refers to converting some elements of malware into images' form for detection. In 2011, Nataraj et al. [13] first proposed to visualize binary files of computer malware as grayscale images and then to classify the malware by using the image features. McLaughlin et al. [14] extracted the opcode sequence of the code instructions in the Android software and mapped it in the vector space and then used CNN to achieve detection. Kumar et al. [15] transformed malware into grayscale scale images and then detected and classified the malware by Random Forest algorithm (RF). On this basis, Darus et al. [16] compared the accuracy of the K-Nearest Neighbor algorithm (KNN), RF, and Decision Tree algorithm (DT) applied to malware grayscale images' classification and detection. And they found out the one with the best performance among the three algorithms is RF. Poonguzhali et al. [17] proposed to combine convolutional neural network and intelligent optimization algorithm for malware image detection and achieved high detection accuracy. Many research based on visualization technology has been implemented in computer malware detection, but few on Android malware detection. The detailed analysis of the above references can be found in Table 1.

There are a lot of types and variations of Android malware; their variations and code obfuscation, shell and encryption, make detection more difficult. Traditional convolutional neural network for Android has many parameters and a large amount of calculation, which leads to a long training time and is not suitable for the mobile Internet platform. For the reasons mentioned above, this study proposes to convert Android malware to RGB images and to train and detect it by depthwise separable convolutional neural networks, which improves the training speed and reduces the number of parameters while having fine accuracy.

The rest of this paper is organized as follows. Section 2 describes the basic principle of convolution neural network

and depthwise separable convolution. In Section 3, it introduces the method of this study which includes a new malware visualization method and MobileNet V2 network model. Experimental results and performance analysis are discussed in Section 4. And the conclusion of this study is presented in Section 5.

## 2. Depthwise Separable Convolutional Neural Networks

*2.1. Deep Learning and Convolutional Neural Networks.* Deep learning [18] is a collection of algorithms that use various machine learning algorithms to solve various problems such as images and texts on multilayer neural networks. It can provide feature learning and obtain higher-level abstract features that human may not recognize. These abstract features can help to capture relevant characteristics. Among deep learning models, convolutional neural networks [19] have outstanding performance in the field of image recognition. Compared with traditional machine learning and other models, CNN has a strong ability to obtain higher-level features of the images. A CNN includes the input layers, the convolution layers, the ReLU excitation layers, the pooling layers, and the fully connected layers. The current common CNN models are VGGNet, ResNet, GoogleNet, and MobileNet.

*2.2. Standard Convolution.* Standard convolution uses  $N \times D_k \times D_k$  convolution kernels to perform convolution calculations on three-channel RGB images and finally output  $N$  Feature Maps. The process is shown in Figure 1.

The number of parameters and calculation cost of standard convolution are described as follows:

$$\begin{aligned} P_1 &= D_k \times D_k \times M \times N, \\ C_1 &= D_k \times D_k \times M \times N \times D_F \times D_F, \end{aligned} \quad (1)$$

where  $D_k \times D_k$  represents the size of the convolution kernel,  $D_k \times D_k$  is the size of the feature maps, and  $M$  and  $N$  denote the number of input channels and output channels, respectively.

*2.3. Depthwise Separable Convolution.* Depthwise separable convolution [20]-based MobileNet v2 model is used in this study. Compared with standard convolution, this model has faster calculation speed and lower computational cost. Depthwise separable convolution is a combination of depthwise convolution (DW) and pointwise convolution (PW), in which the number of parameters and calculation cost of extracting feature are lower than standard convolution.

- (1) Depthwise convolution: after inputting a three-channel RGB image, DW will perform the first convolution operation, which is completely performed in a two-dimensional plane. The number of convolution kernels is the same as the number of channels of the input image. Therefore, a three-

TABLE 1: Analysis of related works.

Types	References	Target features	Methods	Analysis
Static detection	[4]	API call sequence	Decompiling the source code to extract features and detecting them by the probability discriminant model	The pretreatment process is complex
	[6]	Classes and class dependencies	Analyzing the heterogeneity of android malware code by machine learning algorithm	Model training takes a long time
	[7]	Functions call sequence	Computing malware similarity through machine learning algorithm	The accuracy is not high
Dynamic detection	[9]	Malicious behavior	It can be judged by the abnormal behavior of software running	Serious memory consumption
	[10]	Malicious behavior	The behavior features are transformed into hot coding and transmitted to the DBN network for training	The preprocessing is complex and the trigger mechanism needs to run the system all the time
	[11]	System call behavior	Monitor and analyze system call behavior from time to time	The trigger mechanism needs to run the system all the time
	[12]	API functions	Introducing hooks in order to trace restricted API functions used at runtime of the application	The system starts slowly and occupies a large proportion of memory
Visualization detection	[13]	Binary files	Malware binaries are visualized as grayscale images and classified by analyzing the image feature descriptor	The feature of the grayscale image is single and model training takes a long time
	[15]	.Dex files	The .dex files are transformed into RGB images and classified by random forest algorithm	The accuracy is not high
	[17]	Binary files	Converting the malicious code into the grayscale images, these images were identified and classified by the convolutional neural networks; then, using the bat algorithm to eliminate over fitting of the model over the fitting and the under-fitting	The feature of the grayscale image is single and model training takes a long time

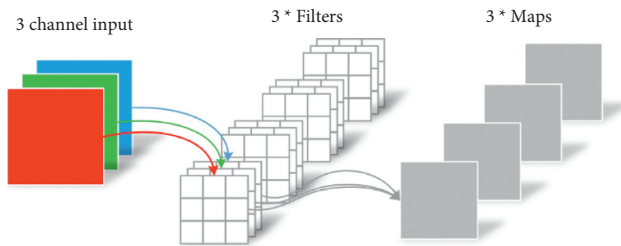


FIGURE 1: Standard convolution process.

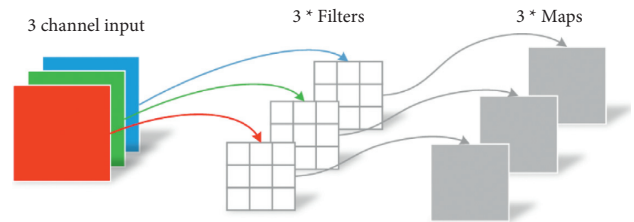


FIGURE 2: Depthwise convolution process.

channel image generates three feature maps after calculation, as shown in Figure 2.

- (2) Pointwise convolution: the operation of the PW is similar to the standard convolution operation, as shown in Figure 3. The size of its convolution kernel is  $1 \times 1 \times M$ , and  $M$  is the number of channels in the previous layer. This convolution will weight and combine the previous maps in the depth direction to generate new feature maps.

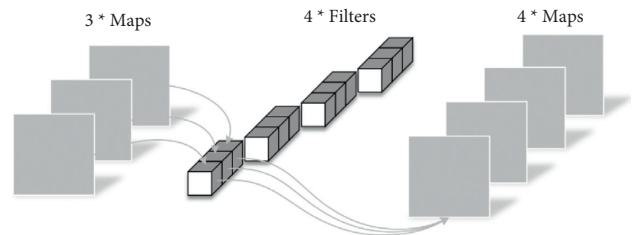


FIGURE 3: Pointwise convolution process.

The number of parameters and calculation cost of the depth separable convolution are represented as follows:

$$\begin{aligned}
 C_2 &= D_k \times D_k \times M \times D_F \times D_F + M \times N \times D_F \times D_F, \\
 P_2 &= M \times D_k \times D_k + M \times N.
 \end{aligned}
 \tag{2}$$

The results of the depthwise convolution and pointwise convolution mentioned above are the same as the result of the

standard convolution. Equations (3) and (4) are the ratios of the number of parameters and the calculation cost of the depthwise separable convolution and the standard convolution under the same effect. It can be seen that the number of parameters and the calculation cost of the depthwise separable convolution are greatly reduced. The size of the convolution kernel is set as  $3 \times 3$  usually. In this case, the number of

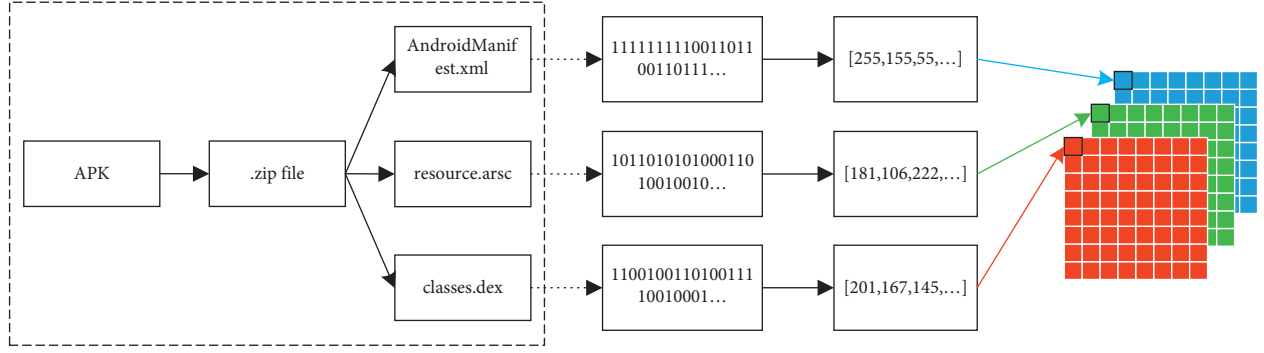


FIGURE 4: Visualization process of Android software.

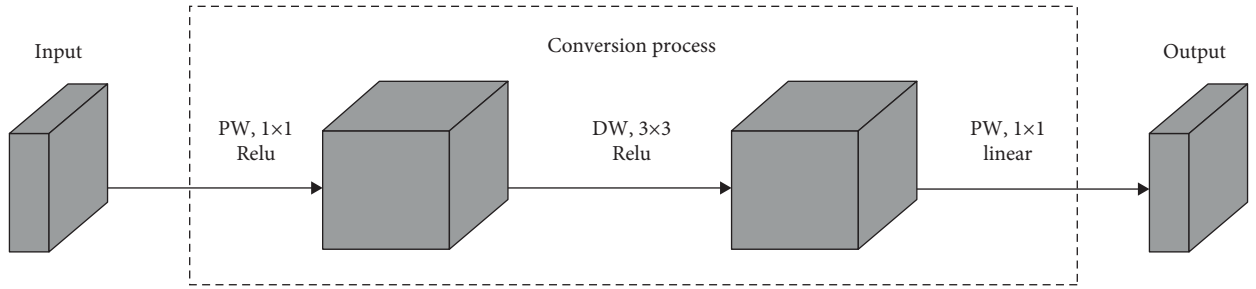


FIGURE 5: Convolution process of inverse residual depthwise separable convolution.

TABLE 2: Comparison of CNN model parameters.

Model name	Number of parameters
VGG16	138342976
Inception v3	24734048
ResNet	23518273
MobileNet v1	4209088
MobileNet v2	2958440

TABLE 3: Classification of dataset (a) and (b).

Dataset	Category	Quantity
(a)	Benign	13500
	Malware	13787
(b)	Adware	1515
	Banking	2506
	Benign	4942
	Riskware	4362
	SMS malware	4822

parameters and calculation cost of the depthwise separable convolution are about 1/9 of those of the standard convolution:

$$\frac{P_2}{P_1} = \frac{D_k \times D_k \times M + M \times N}{D_k \times D_k \times M \times N} = \frac{1}{N} + \frac{1}{D_k^2}, \quad (3)$$

$$\begin{aligned} \frac{C_2}{C_1} &= \frac{D_k \times D_k \times M \times D_F \times D_F + M \times N \times D_F \times D_F}{D_k \times D_k \times M \times N \times D_F \times D_F} \\ &= \frac{1}{N} + \frac{1}{D_k^2}. \end{aligned} \quad (4)$$

### 3. Methods

In this section, this study primarily proposes to visualize Android software as RGB images and then detects it through the lightweight convolution neural network.

**3.1. Visualization Process.** Android application package (APK) [21] is a file format used by the Android system to install applications (APP), and the suffix is .apk. In fact, the APK can be decompressed by changing the suffix name to .zip, which contains META-INF folder, res folder, resources.arsc, AndroidManifest.xml, and classes.dex. Among them, classes.dex is an 8 bit binary file, which is the logical part of the Android software to realize functions. Resources.arsc is the compiled binary resource file. AndroidManifest.xml contains the configuration information of the APP.

This study proposes to convert Android software to RGB images for detection. RGB [21] images are composed of a three-dimensional array of the  $M \times N \times 3$  format, which can also be understood as being constructed by three  $M \times N$  grayscale images. The three images represent the three components of R, G, and B. And the pixel point of each component ranges from [0, 255]. From this, the value expression of RGB image pixel points is (R: [0, 255], G: [0, 255], B: [0, 255]). The visualization process is shown in Figure 4. The suffix name of the APK is changed to .zip file. Then, the classes.dex, resources.arsc, and AndroidManifest.xml are extracted by the zip file library of the python. The decimal range of each byte is in [0, 255],

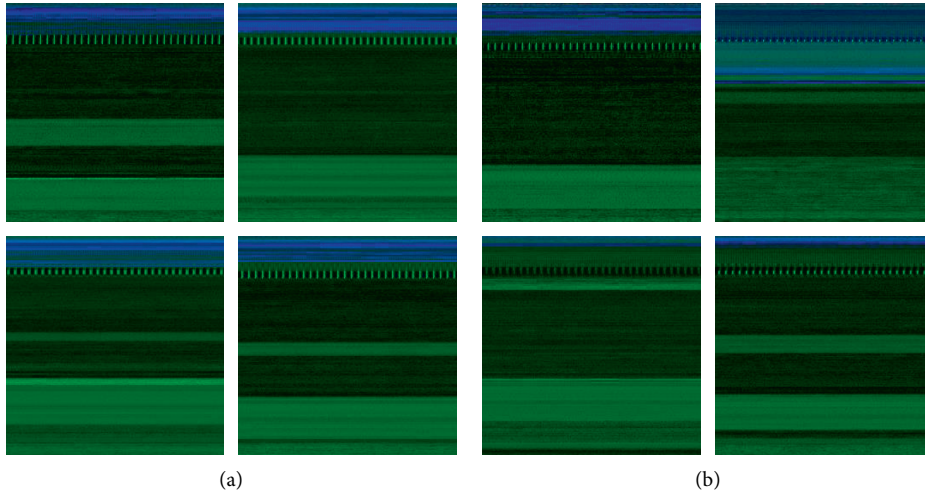


FIGURE 6: Some RGB images of (a) benign software and (b) malware.

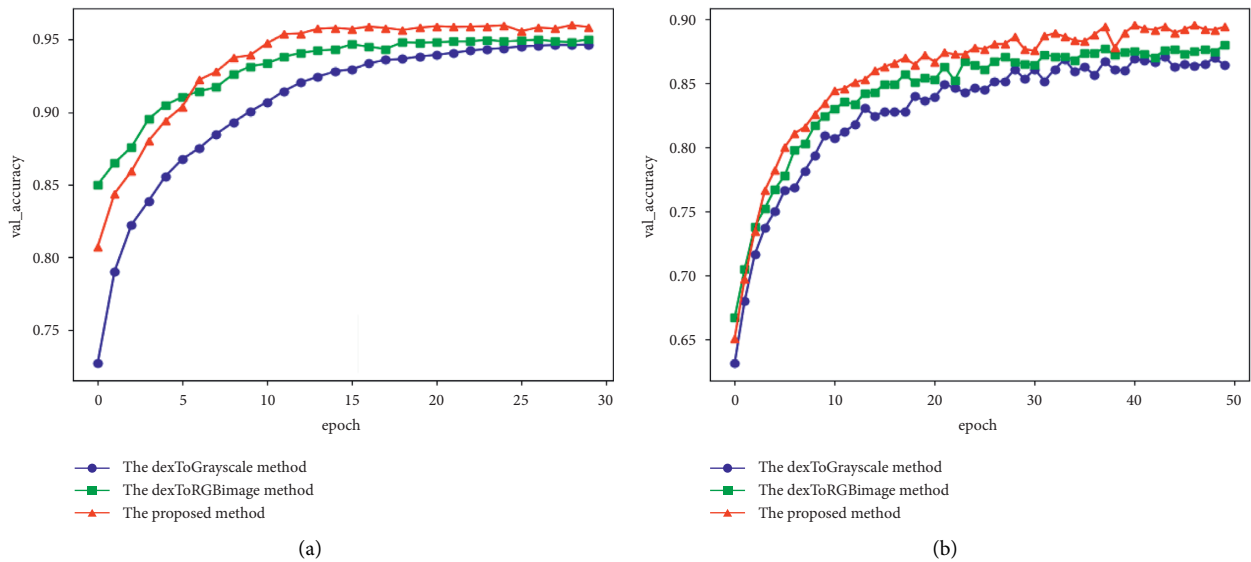


FIGURE 7: Training process of visualization methods.

which is exactly the same as the value range of each component pixel of the RGB images. So, these three files are filled with channels of R, G, and B to generate RGB images by the PIL library.

**3.2. Model Summary.** The model named MobileNet v2 [22] implemented in this study is a lightweight CNN model proposed by Google in 2018, which uses depthwise separable convolution. MobileNet v2 is an improvement in MobileNet v1. MobileNet v1 is only a stack of depthwise separable convolutions; however, MobileNet v2 introduces a residual structure to improve the performance of the network. In addition, in order to preserve the diversity of features, the nonlinear activation function of ReLU is not adopted at the end of each depthwise separable convolution. The structure of MobileNet includes DW and PW. Before DW, MobileNet v2 adds a PW for performing dimension upgrade. This

method makes DW work in higher dimensions, which can get a better effect. The convolution structure of MobileNet v2 is shown in Figure 5.

Table 2 shows the number of parameters of MobileNet v2 and other models. It can be seen that MobileNet v2 has fewer parameters and faster training speed than MobileNet v1, so this study uses MobileNet v2 for Android malware detection.

**3.3. Functions of Training Process.** Android malware detection is a two-classification problem. Sigmoid which is a good threshold function is used as the output of the classifier. It can map the output variable to (0, 1). If the output variable is greater than 0.5, it is considered to belong to one category; otherwise, it belongs to another category. The formula for the sigmoid function is as follows:

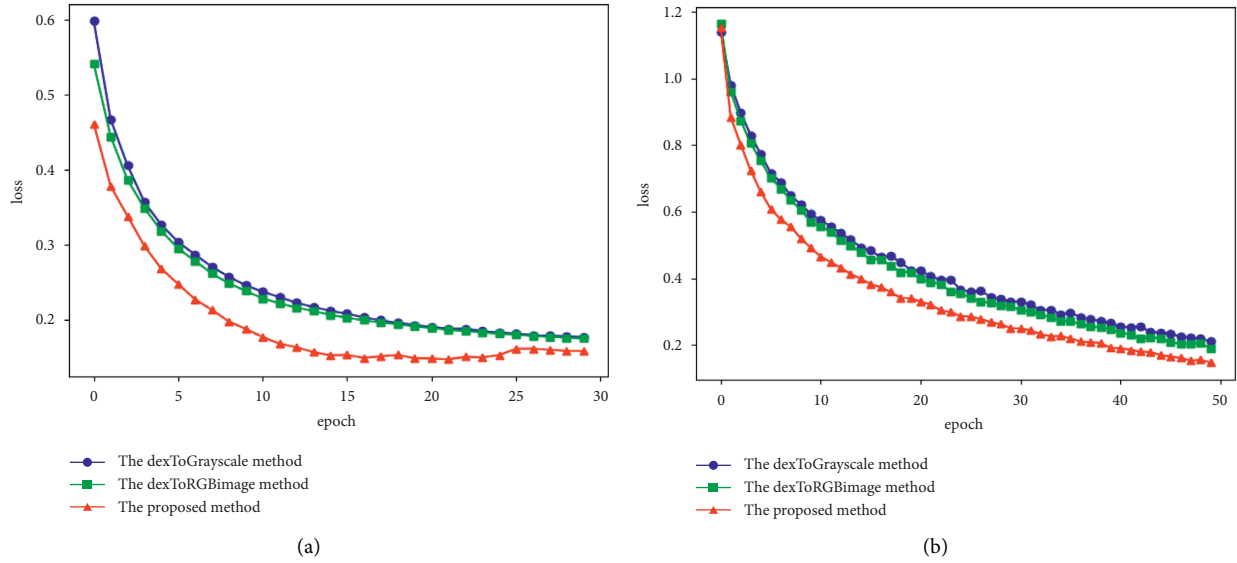


FIGURE 8: Loss-epoch curves of visualization methods.

TABLE 4: Comparison of accuracy and training time of visualization methods.

Dataset	Neural networks' models	Visualization methods	Val_accuracy (%)	Training time (1 epochs)
((s)a)	MobileNet v2	dexToGrayscale	94.44	42.02
		dexToRGBImage	95.01	44.41
		<b>The proposed method</b>	<b>95.98</b>	46.67
(b)	MobileNet v2	dexToGrayscale	87.09	39.99
		dexToRGBImage	87.97	40.18
		<b>The proposed method</b>	<b>89.54</b>	41.44

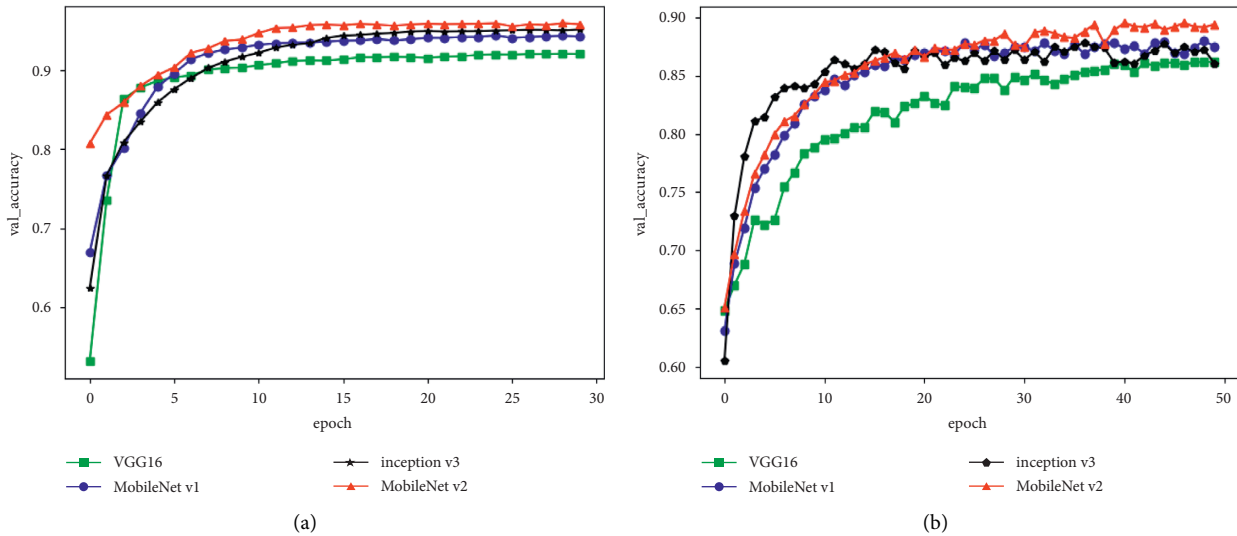


FIGURE 9: Training process of CNN models.

$$\text{sigmoid}(x) = \frac{1}{1 + e^{-x}}, \quad (5)$$

where  $x$  represents the score of the sample of this category in the neural network.

Binary\_cross entropy is used as the loss function in the training process of CNN, which is described as follows:

$$\text{loss} = - \sum_{i=1}^n \hat{y}_i \log y_i + (1 - \hat{y}_i) \log (1 - y_i), \quad (6)$$

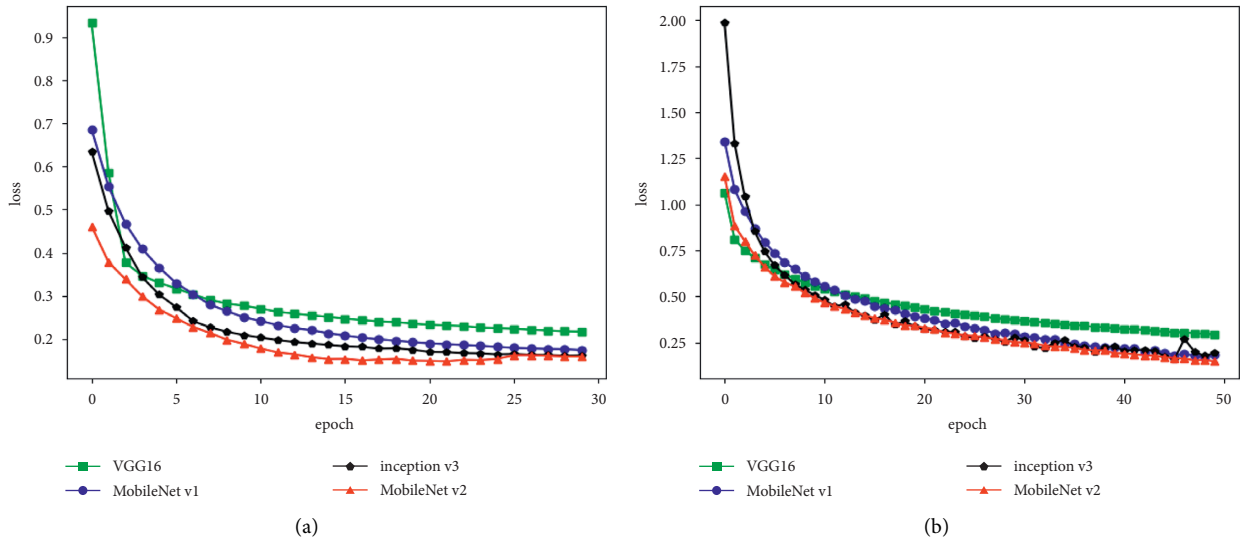


FIGURE 10: Loss-epoch curves of visualization methods.

TABLE 5: Comparison of accuracy and training time of CNN models.

Dataset	Visualization methods	Neural network models	Val_accuracy (%)	Training time (1 epochs)
((s)a)	<b>The proposed method</b>	Model 1: VGG16 [17]	92.12	74.11
		Model 2: MobileNet v1	94.39	46.03
		Model 3: Inception v3 [23]	95.19	62.55
		<b>The proposed model: MobileNet v2</b>	<b>95.98</b>	46.67
(b)	<b>The proposed method</b>	Model 1: VGG16 [17]	86.22	71.01
		Model 2: MobileNet v1	88.01	40.87
		Model 3: Inception v3 [23]	87.78	58.14
		<b>The proposed model: MobileNet v2</b>	<b>89.54</b>	41.44

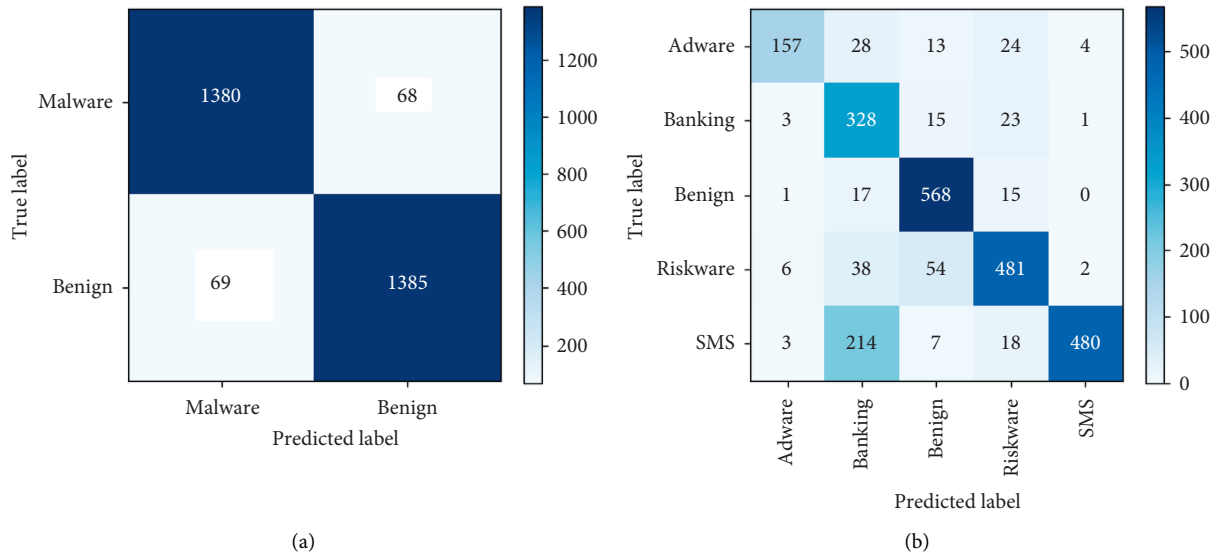


FIGURE 11: Confusion matrix for dataset (a) and (b).

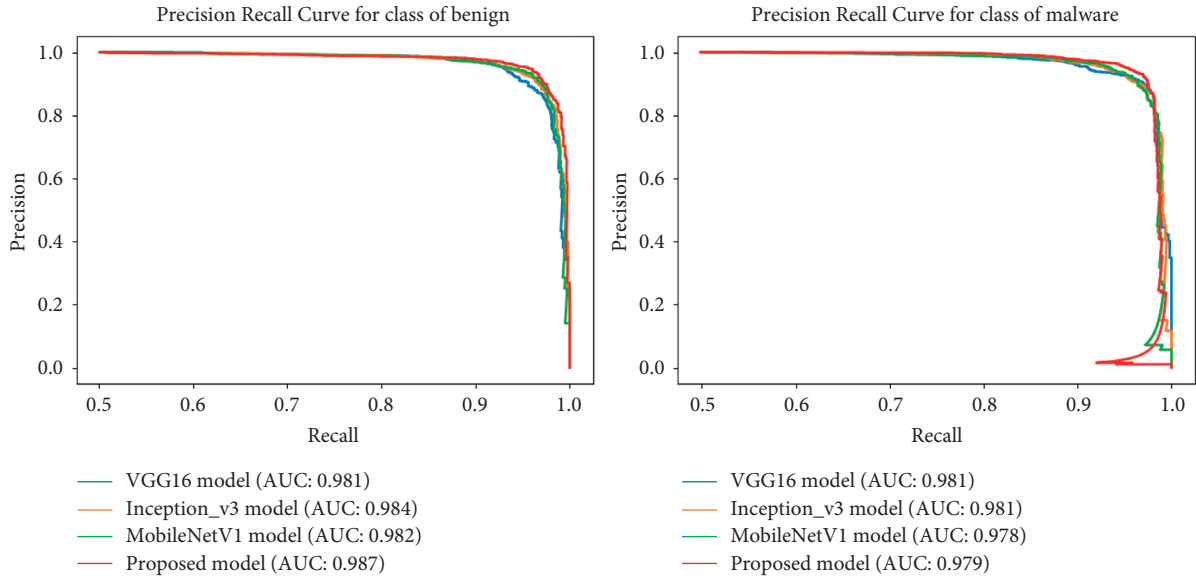


FIGURE 12: The P-R curves of dataset (a).

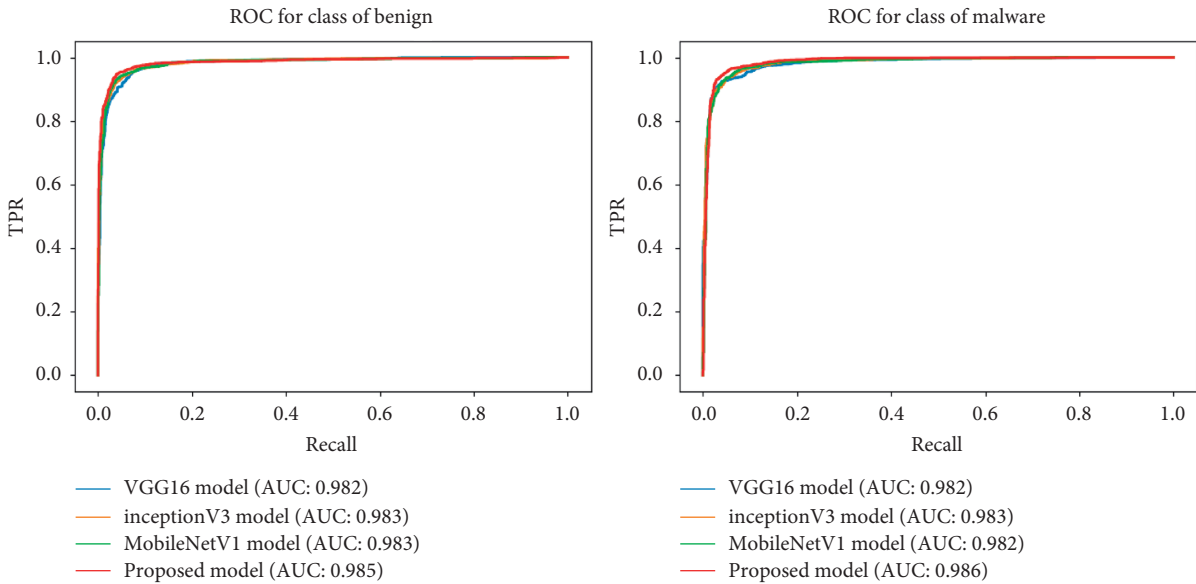


FIGURE 13: The ROC curves of dataset (a).

where  $\hat{y}_i$  is the true label value (0 or 1) and  $y_i$  is the predicted value given by the neural network model. At the end of each single training session, the neural networks will calculate the loss and then update the model parameters according to the loss.

#### 4. Experiments and Discussion

This experiment was carried out under Ubuntu 18.04 version, the CPU is 8-core E5-2687W, the memory is 16 GB, the GPU is NVIDIA RTX 2080 TI, and the memory is 11 GB. This study use python and Tensorflow2.0 framework to design CNN models.

**4.1. Dataset.** The following two datasets were used in this experiment, and the details are shown in Table 3.

- (1) Dataset (a) contains CIC-AndMal-2017, CIC-AndMal-2020, and 3000 benign Android software collected by our group. This dataset is divided into benign software and malware.
- (2) Dataset (b) is CIC-AndMal-2020, which contains 5 types of Android software.

The dataset of this experiment has not been found used in other documents before. And this dataset is adopted in this study to compare the performance of the methods based on different visualization methods and different CNN models.

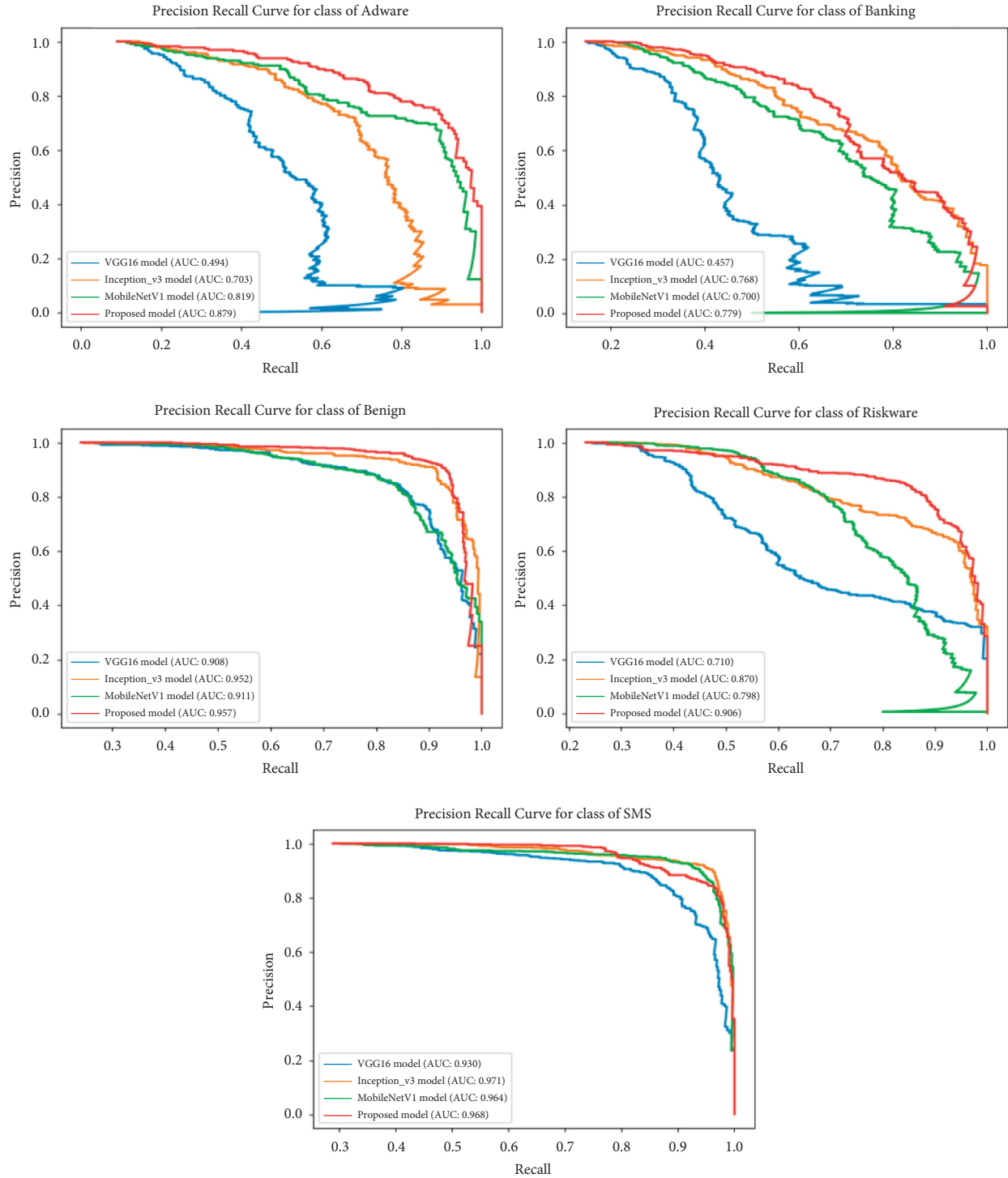


FIGURE 14: The P-R curves of dataset (b).

4.2. Comparison of Visualization Methods. In this study, dataset (a) and dataset (b) are separately divided into a training set and a test set of eight to two, and all samples are visualized to RGB images by the method mentioned in Section 3.1. Then, all images are scaled to the same size. Figure 6 shows some  $256 \times 256$  RGB images generated by randomly selected Android benign software and malware. The difference in image texture between benign software and

malware is very minor. However, the depthwise separable convolutional neural networks have a strong ability to image recognition and can judge whether it is malware through the subtle difference in the images.

The five-fold cross-validation method which can improve the accuracy and stability of the model is used in this experiment. The network parameters are initialized by MobileNet v2. Except for the last 10 layers of training, all the



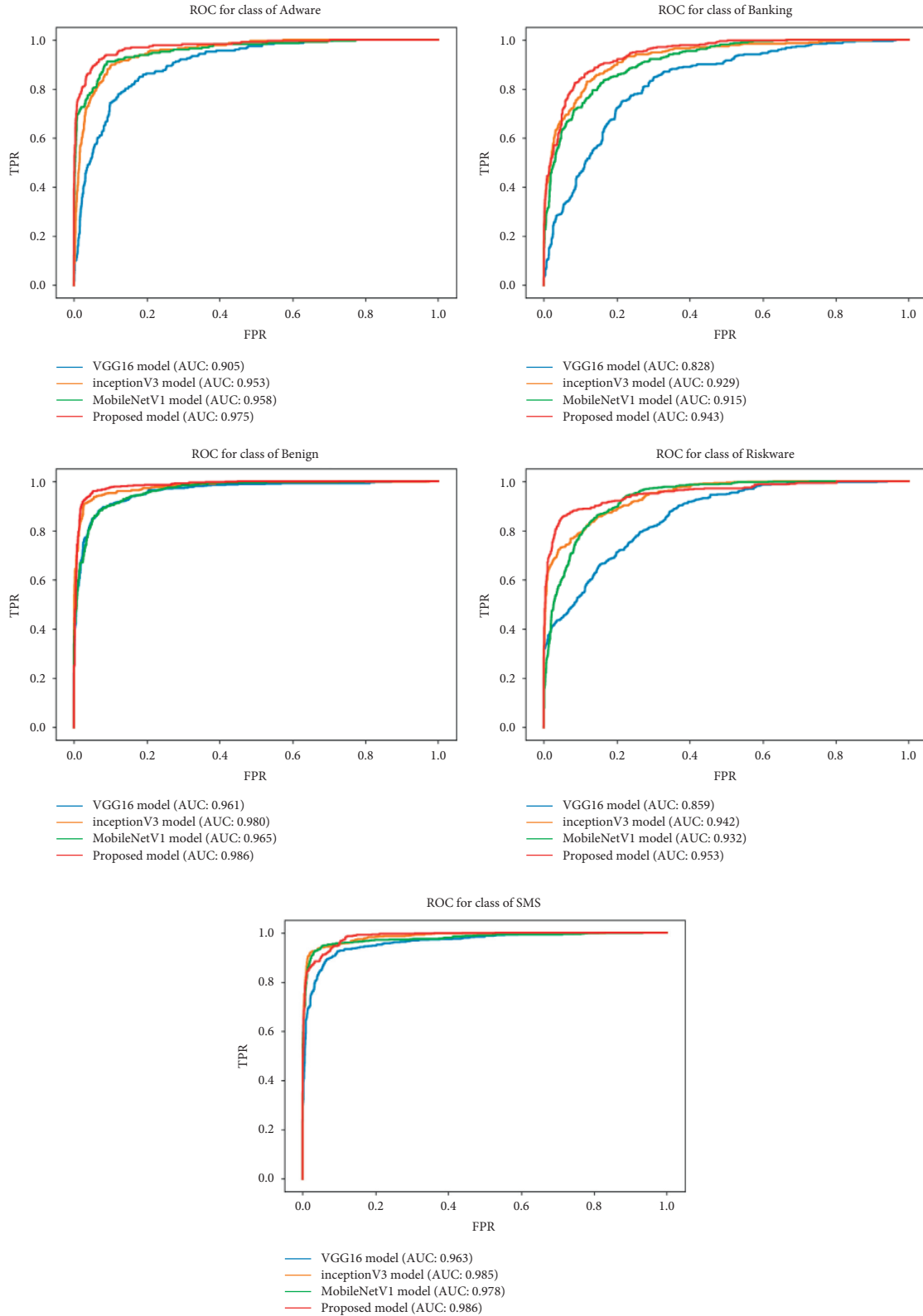


FIGURE 15: The ROC curves of dataset (b).

front layers are frozen. In the training process, the initial learning rate is set to 0.00001 and BATCH\_SIZE is set to 32. The epoch is set to 30 in the experiment of dataset (a) and 50 for dataset (b).

Traditional visualization methods of Android software include dexToGrayscale and dexToRGBimage. They refer to the conversion of classes. dex files in APK into grayscale images or RGB images. The RGB images which are generated

by the classes.dex, resources.arsc, and AndroidManifest.xml can be easily distinguished in convolutional neural networks. The proposed visualization method is compared with the traditional visualization methods in the MobileNet v2 network model. The experiment is shown in Figures 7 and 8 and Table 4.

According to the results, the dexToGrayscale method needs more epochs to steady state and has the lowest accuracy. The effect of the dexToRGBimage method is better than the dexToGrayscale method. And the proposed method has the highest accuracy in the similar training time.

**4.3. Comparison of Models.** In this study, the CNN models in the previous document are compared with the MobileNet v2 used in this project, such as VGG16, Inception v3, and MobileNet v1.

The training process for dataset (a) and (b) is shown in Figures 9 and 10. The accuracy and the training time of the four models under the same conditions can be seen from Table 5.

In the training of dataset (a), the VGG16 model gets the lowest accuracy of 92.12% and the slowest training speed rate of 74.11 seconds at 1 epoch. The MobileNet v1 has fast training spend rate and high accuracy. And the Inception v3 is the opposite of it. The proposed method with MobileNet v2 has the accuracy of 95.98% and the training speed rate of 41.44 seconds at 1 epoch, which gets the highest accuracy in a relatively short time and has a certain improvement compared with the methods in historical document. Training result of dataset (b) is like that of dataset (a), MobileNet v2 takes the accuracy rate of 89.54%, and the training speed rate is 41.44 seconds at 1 epoch.

**4.4. Model Analysis.** Figure 11 shows the confusion matrix for dataset (a) and (b). In dataset (a), the true positive rate and false negative rate reached 99.3% and 95.2%, respectively. Among them, the recall rate of malicious software is 95.1%. In dataset (b), because of the small number of samples, many kinds and data imbalance, detection results of adware and SMS are not very good, but the true positive rate of banking malware, benign software, and riskware achieved 88.6%, 94.5%, and 82.8%, respectively.

Figures 12 and 13 show the P-R curves of dataset (a) and (b). The P-R curve is drawn under certain some thresholds based on precision rate and recall rate, which is an evaluation indicator that shows the quality of the model. In dataset (a), the AUC of malware and benign is 0.978 and 0.982, respectively. In another dataset, AUC of the five categories is 0.839, 0.779, 0.957, 0.906, and 0.968.

Figures 14 and 15 show the ROC curves of dataset (a) and (b). The abscissa of the ROC curve is the false positive rate and the ordinate is true positive rate. And ROC curve is a standard to measure the quality of a classification model. As can be seen from the figures, the model performs better in the margin class and SMS class because the number of other three kinds of data is small. Overall, the MobileNet V2 model used in this study has good performance and stability.

**4.5. Discussion.** Through the above experiments, it can be seen that the malware visualization method introduced in this study is better than the methods in [17, 23], etc. And its generalization ability is good. The performance and efficiency of MobileNet V2 in this experiment are also superior to those of the conventional convolutional neural network model. However, the model memory consumption of MobileNet V2 is still large. For the forthcoming period, we will consider compressing the network structure by referring to other lightweight network models, such as ShuffleNet [24], or optimizing the feature quantity by combining three feature files to generate a Markov graph [25] to further improve the performance of the detection method.

## 5. Conclusions

Android malware is continuously causing safety hazards to people's life, so the detection of Android malware is very necessary. This study proposed an Android malware detection method based on depthwise separable convolutional neural networks. The proposed algorithm converts classes.dex, resources.arsc, and AndroidManifest.xml of the Android malware samples to the RGB image. Compared with the grayscale image, RGB image possesses better color and texture characteristics. The quality and accuracy of MobileNet v2 model used in this experiment are higher than other models. Moreover, the number of parameters and calculation cost of MobileNet v2 are far lower than other models. If the model is deployed on Android phones, it can achieve a relatively outstanding malware detection function of the case of occupying a small amount of memory. So, this model is more suitable for Android malware detection. The significant improvement of the proposed method has been demonstrated by comparing it with the methods of some historical research.

## Data Availability

Restrictions are applied to the availability of these data. Data were obtained from Canadian Institute for Cybersecurity and are available at <https://www.unb.ca/cic/datasets/andmal2020.html> with the permission of Canadian Institute for Cybersecurity.

## Conflicts of Interest

The authors declare that they have no conflicts of interest.

## References

- [1] The AV-Test, "Facts & Analyses on the Threat Scenario: The AV-TEST Security Report 2019/2020," pp. 4–10, 2020, [https://www.av-test.org/fileadmin/pdf/security\\_report/AV-TEST\\_Security\\_Report\\_2019-2020.pdf](https://www.av-test.org/fileadmin/pdf/security_report/AV-TEST_Security_Report_2019-2020.pdf).
- [2] Q. Han, V. S. Subrahmanian, and Y. Xiong, "Android malware detection via (somewhat) robust irreversible feature transformations," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 3511–3525, 2020.
- [3] Y. Feng, S. Anand, I. Dillig, and A. Aiken, "Apposcopy: Semantics-based detection of Android malware through static analysis," in *Proceedings of the 22nd ACM SIGSOFT*

- International Symposium on Foundations of Software Engineering, Association for Computing Machinery*, pp. 576–587, Hong Kong, China, November 2014.
- [4] L. Cen, C. S. Gates, L. Si, and N. Li, “A probabilistic discriminative model for android malware detection with decompiled source code,” *IEEE Transactions on Dependable and Secure Computing*, vol. 12, no. 4, pp. 400–412, 2015.
  - [5] L. Onwuzurike, E. Mariconti, P. Andriotis, E. De Cristofaro, G. Ross, and G. Stringhini, “MaMaDroid: Detecting android malware by building Markov chains of behavioral models (extended version),” *ACM Trans. Priv. Secur.*, vol. 22, no. 2, p. 34, 2019.
  - [6] K. Tian, D. Yao, B. G. Ryder, G. Tan, and G. Peng, “Detection of repackaged android malware with code-heterogeneity features,” *IEEE Transactions on Dependable and Secure Computing*, vol. 17, no. 1, pp. 64–77, 2020.
  - [7] D. Kong and G. Yan, “Discriminant malware distance learning on structural information for automated malware classification,” *ACM SIGMETRICS - Performance Evaluation Review*, vol. 41, no. 1, pp. 347–348, 2013.
  - [8] B. Yu, Y. Fang, Q. Yang, Y. Tang, and L. Liu, “A survey of malware behavior description and analysis,” *Frontiers of Information Technology & Electronic Engineering*, vol. 19, no. 5, pp. 583–603, 2018.
  - [9] A. Saracino, D. Sgandurra, G. Dini, and F. Martinelli, “MADAM: Effective and efficient behavior-based android malware detection and prevention,” *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 1, pp. 83–97, 2018.
  - [10] 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.
  - [11] M. Zheng, M. Sun, and J. C. S. Lui, “DroidTrace: A ptrace based Android dynamic analysis system with forward execution capability,” in *Proceedings of the 2014 International Wireless Communications and Mobile Computing Conference (IWCMC)*, pp. 128–133, Nicosia, Cyprus, August 2014.
  - [12] O. Somarriba, U. Zurutuza, R. Uribeetxeberria, L. Delosières, and S. Nadjm-Tehrani, “Detection and visualization of android malware behavior,” *JECE*, vol. 2016, p. 6, 2016.
  - [13] L. Nataraj, S. Karthikeyan, G. Jacob, and B. S. Manjunath, “Malware images: Visualization and automatic classification,” in *Proceedings of the 8th International Symposium on Visualization for Cyber Security, Association for Computing Machinery*, p. 4, Pittsburgh, Pennsylvania, USA, JULY 2011.
  - [14] N. McLaughlin, J. M. d. Rincon, B. Kang et al., “Deep android malware detection,” in *Proceedings of the Seventh ACM on Conference on Data and Application Security and Privacy, Association for Computing Machinery*, pp. 301–308, Scottsdale, Arizona, USA, 2017.
  - [15] A. Kumar, K. P. Sagar, K. S. Kuppusamy, and G. Aghila, “Machine learning based malware classification for Android applications using multimodal image representations,” in *Proceedings of the 2016 10th International Conference on Intelligent Systems and Control (ISCO)*, pp. 1–6, Coimbatore, India, January 2016.
  - [16] F. M. Darus, S. N. A. Ahmad, and A. F. M. Ariffin, “Android malware detection using machine learning on image patterns,” in *Proceedings of the Cyber Resilience Conference*, Putrajaya, Malaysia, November 2018.
  - [17] N. P. Poonguzhali, T. Rajakamalam, S. Uma, and R. Manju, “Identification of malware using CNN and bio-inspired technique,” in *Proceedings of the 2019 IEEE International Conference on System, Computation, Automation and Networking (ICSCAN)*, pp. 1–5, Pondicherry, India, March 2019.
  - [18] G. E. Hinton, S. Osindero, and Y.-W. Teh, “A fast learning algorithm for deep belief nets,” *Neural Computation*, vol. 18, no. 7, pp. 1527–1554, 2006.
  - [19] J. Gu, Z. Wang, J. Kuen et al., “Recent advances in convolutional neural networks,” *Computer Science*, vol. 1, 2015.
  - [20] F. Chollet, “Xception: Deep learning with depthwise separable convolutions,” in *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1800–1807, Honolulu, HI, USA, July 2017.
  - [21] S. Zia, B. Yüksel, D. Yüret, and Y. Yemez, “RGB-D object recognition using deep convolutional neural networks,” in *Proceedings of the 2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, pp. 887–894, Venice, Italy, October 2017.
  - [22] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. Chen, “MobileNetV2: Inverted residuals and linear bottlenecks,” in *Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4510–4520, Salt Lake City, UT, USA, June 2018.
  - [23] T. H. Huang and H. Kao, “R2-D2: ColoR-inspired convolutional NeuRal network (CNN)-based Android malware detections,” in *Proceedings of the 2018 IEEE International Conference on Big Data (Big Data)*, pp. 2633–2642, Seattle, WA, USA, December 2018.
  - [24] X. Zhang, X. Zhou, M. Lin, and J. Sun, “ShuffleNet: An extremely efficient convolutional neural network for mobile devices,” in *Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, USA, June 2018.
  - [25] H. Divandari, B. Pechaz, and M. Vafaie, “Malware detection using Markov blanket based on opcode SequencesMalware detection using Markov blanket based on opcode sequences,” in *Proceedings of the 2015 International Congress on Technology, Communication and Knowledge (ICTCK)*, Mashhad, Iran, November 2016.

## Research Article

# 5G and Blockchain Enabled Lightweight Solutions for Containing COVID-19

Mohsin Kamal <sup>1</sup>, Abdulah Jeza Aljohani <sup>2,3</sup>, Eisa Alanazi,<sup>4</sup> and Fahad R. Albogamy<sup>5</sup>

<sup>1</sup>KIOS Research and Innovation Center of Excellence, University of Cyprus, Nicosia 1678, Cyprus

<sup>2</sup>Department of Electrical and Computer Engineering, King Abdulaziz University, Jeddah 21589, Saudi Arabia

<sup>3</sup>Center of Excellence in Intelligent Engineering Systems (CEIES), King Abdulaziz University, Jeddah 21589, Saudi Arabia

<sup>4</sup>University of Umm Al-Qura, Makkah, Saudi Arabia

<sup>5</sup>Computer Sciences Program, Turabah University College, Taif University, P.O. Box 11099, Taif 21944, Saudi Arabia

Correspondence should be addressed to Mohsin Kamal; [kamal.mohsin@ucy.ac.cy](mailto:kamal.mohsin@ucy.ac.cy)

Received 6 August 2021; Accepted 3 December 2021; Published 15 February 2022

Academic Editor: Muhammad Ahmad

Copyright © 2022 Mohsin Kamal et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Due to the global pandemic of COVID-19, there is an urgent need to utilize existing technologies to their full potential. Internet of things (IoT) is regarded as one of the most trending technologies with a great potential in fighting against the coronavirus outbreak. In this study, we examine the current status of IoT applications related to COVID-19, identify their deployment and operational challenges, and suggest possible opportunities to further contain the pandemic. Furthermore, we perform analysis by examining the IoT implementation in which internal and external factors are discussed. We suggest by presenting results that lightweight security algorithms, blockchain-based solutions for enabling end-to-end security and privacy, and 5G for IoT devices to tackle the bandwidth issues for scalable IoT networks are few of the solutions in containing the COVID-19.

## 1. Introduction

The Internet of things (IoT) consists of a complex network of smart devices that frequently exchange data over the Internet [1]. It has renovated the actual world objects into clever virtual objects. The goal of IoT is to unite everything in our world under a mutual arrangement, helping the users in not only controlling the objects around them but also keeping them up to date about the state of things [2]. IoT devices sense the environment and send the acquired data to the Internet cloud without the requirement of human-to-human or human-to-machine interaction. IoT has become an integral part of today's modern era of communication where tens of millions of devices are connected via IoT and the number is growing rapidly [3].

IoT has the potential to play a vital role in various fields of life, such as health systems [4], autonomous vehicles [5], home and industrial automation [6], intelligent transportation [7], and smart grids [8]. Sensors obtain data of related information from the environment and use the

Internet cloud as a medium of delivering information to the relevant body or organization [9]. The core concept behind IoT is the realization of multiple devices communicating with each other seamlessly. This has the promise of better utilization of available resources, reduction in cost, and minimizing manual interaction. As the 2019 coronavirus disease (COVID-19) continues to spread across the globe, it is inevitable to discuss and articulate the IoT potential during pandemics. As of August 6, 2021, the number of COVID-19-confirmed cases has exceeded 80 million [10]. Researchers from different fields continue to investigate and generate diverse solutions, which could help in combating the COVID-19 [11].

IoT comes up with the ingredients needed to help the countries in minimizing the effect of COVID-19. IoT has a wide range of applications, which would be effective to make sure that all the guidelines of safety and precautions provided by health officials are followed. IoT has a scalable network, which has the potential to deal with huge amount of data received from sensors used by number of

applications to fight against COVID-19. Furthermore, the reliable IoT networks decrease the delivery time of crucial information, which can help in providing timely response during the global pandemic of COVID-19 [12]. The role of IoT was never needed to the extent to which it is required now because of coronavirus outbreak.

The key contributions of this study are as follows:

- (i) This study discusses the effectiveness of IoT in combating the global pandemic of COVID-19
- (ii) Several scenarios are examined in which IoT can help in reducing the outbreak of coronavirus
- (iii) We have analyzed the possible challenges that the IoT-based solutions encounter in combating the coronavirus
- (iv) We present the solutions for the challenges by providing the results

The rest of the study is organized as follows. Section 2 presents the important applications of IoT in the perspective of COVID-19. The challenges in implementing the IoT are described in Section 3. The SWOT analysis is performed in Section 4. The solutions to tackling the challenges in deploying the IoT are presented in Section 5. The study is concluded in Section 6.

## 2. Applications of IoT to Combat COVID-19

The seamless connections and vigorous integration with other technologies have enabled the IoT to be one of the promising technologies that will change our lives [13]. The applications of IoT in combating this global pandemic can be spread to several sectors, which can play a major role in reducing the risk of coronavirus outbreak [14]. Figure 1 shows potential applications in which IoT technologies can be useful and effective in combating the COVID-19. The following subsections will examine the capability of the IoT in fighting against COVID-19.

*2.1. Internet of Health Things and Digital Telehealth.* Internet of Health Things (IoHT) is an application of IoT, which aims to connect patients to healthcare facilities to monitor human body vital signs using communication infrastructure [15]. Telemedicine is getting popular in remote areas where accessibility to a quality physician is limited due to different factors. For example, heart rate, electrocardiography, diabetes, and vital body signs can be remotely monitored without the physical presence of patients [16]. An example of the remote data acquisition using IoHT system is shown in Figure 2. The sensors and actuators receive data from patient and send the information to the cloud using a local gateway. The doctor examines the data using any mobile or desktop application provided to them and notifies the patient or medical staff taking care of the patient about the report [17].

Telehealth can play a very important role during the COVID-19 outbreak [18]. A portal is created where patients interact with the doctors and the treatment is provided remotely. The benefit of employing a secured IoHT system in COVID-19 is that the physicians do not come in direct

contact with the patients, hence avoiding the spread of virus [19]. Many countries have started operating the digital telehealth in this time of crisis. Health Arc [20] is used in the USA, Canada, and United Kingdom, which provides IoT-based healthcare devices to the patients whose data are continuously monitored by the medical staff. The data are analyzed, and the suggestions and prescription are provided to the patients on their mobiles or tablets. ContinuousCare provides services in India [21], HealthNet Connect [22] is used in six different states of the USA, and SehatYab [23] providing services in Pakistan is among the leading telehealth service providers. A person with COVID-19 symptoms can use assessment tool provided on the digital platform such as “COVID-19 Gov PK mobile app” [10] provided by the government of Pakistan, which is accessed by the physicians remotely. Using this tool, patients are timely guided and many precious lives can be saved. Furthermore, it also serves to reduce the number of hospitalizations, readmissions, and density of patients in hospitals, all of which help in improving the quality of life and providing timely treatment to COVID-19 patients.

*2.1.1. IoT-Enabled Ambulances.* Medical staff associated with ambulances are usually dealing with very high-pressure and error-prone situations [24]. During the current pandemic of COVID-19, the situations have become even more tensed and pressurized for medical staff dealing COVID-19 patients. The IoT-aided ambulances offer an effective solution in which remote medical experts suggest necessary actions to the medical staff dealing with the patient in the ambulance. This leads to the timely response and effective handling of patient. Figure 3 shows the smart ambulance, which is equipped with IoT-based technology. WAS vehicles [25] provide smart solution-based emergency vehicles. The radio-frequency identification (RFID)-based equipment is connected to wireless local area network (WLAN). The information of patient is remotely accessible by the concerned medical staff.

*2.1.2. IoT-Enabled Healthcare and Safety Devices.* IoHT-aided equipment is classified into two categories, i.e., personal and clinical [26]. Personal-aided IoHT gadgets are used for self-monitoring of health [27]. The most common gadgets used are Apple watch [28] and Fitbit [29]. The user tracks the heartbeat, exercise, sleep, nutrition, and weight using these gadgets. These are useful in fighting against COVID-19 as well because rest and sleep become very important factors for the patients suffering from this disease. The patient can see his reports on the portals provided by these gadget makers and provide information to the related physicians if required. IoT-based wearable gadgets can help in reducing the spread of coronavirus if certain algorithms are implemented to the existing devices. The wearable devices notify in real time if

- (i) The social distancing protocol is violated
- (ii) Any COVID-19 patient is in the locality
- (iii) The area was declared as danger zone by the government in the perspective of coronavirus outbreak

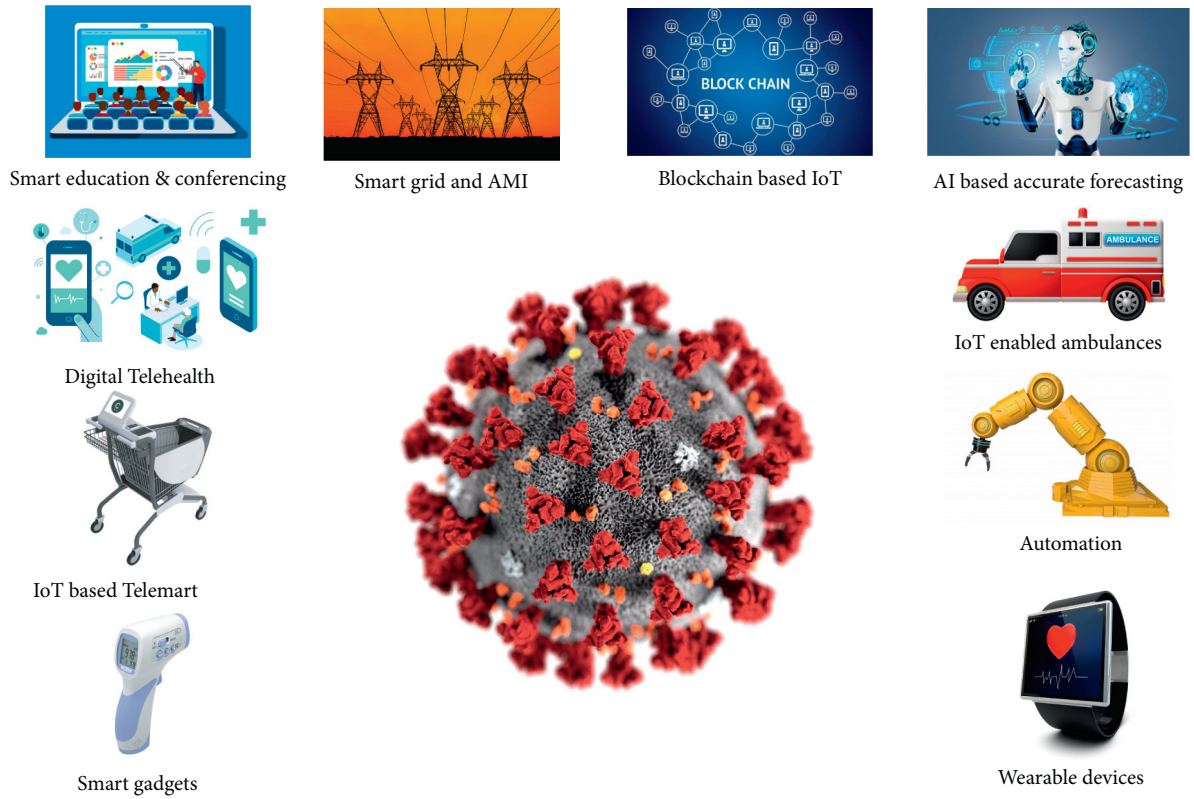


FIGURE 1: Potential IoT applications to combat COVID-19.

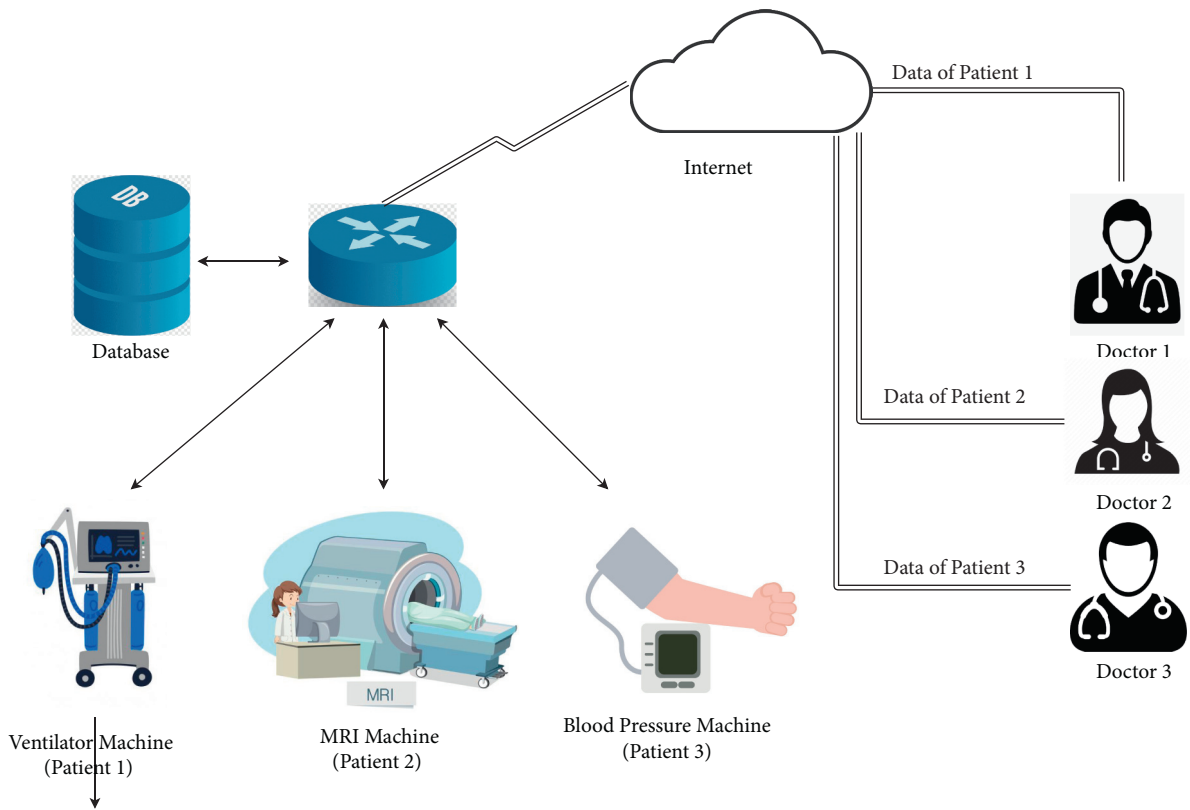


FIGURE 2: Remote examination of medical patients by the doctors in IoT.

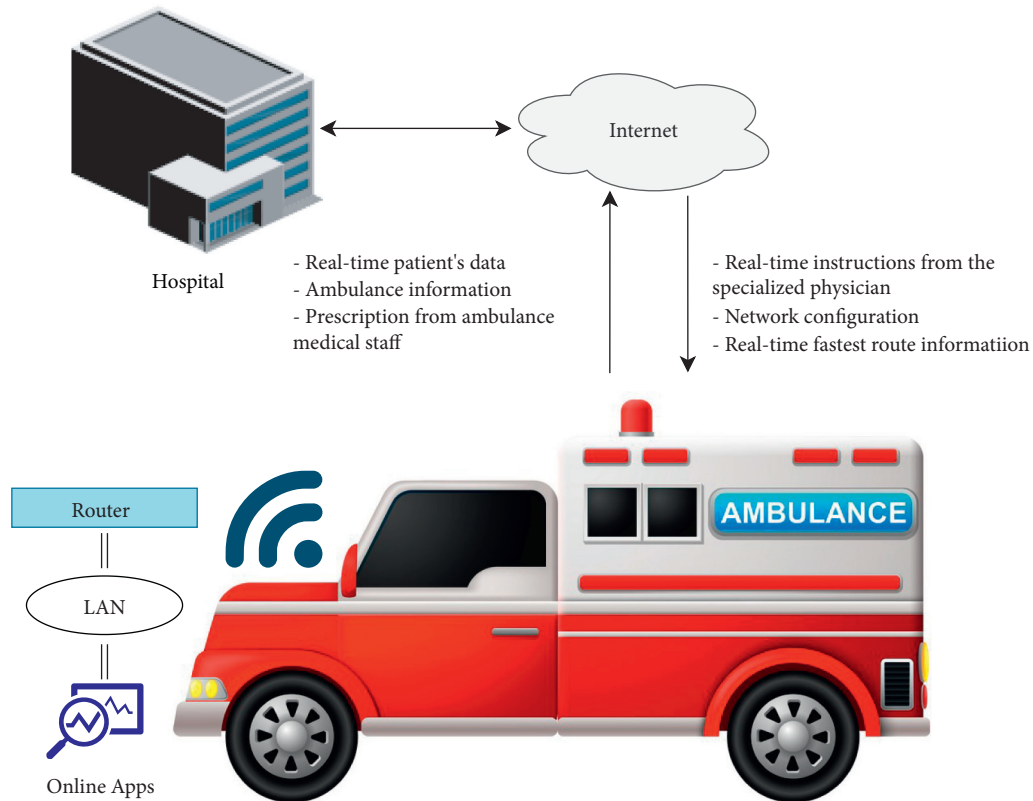


FIGURE 3: IoT-based smart ambulance system.

Figure 4 shows the presence of user in safe zone [10]. If a user moves in the area, which is danger zone with respect to COVID-19 patients, the device intimates in real time and user can maximize precautions to stay safe from coronavirus. Apple and Google have recently partnered in developing a contact tracing technology, which helps in reducing the spread of coronavirus [30, 31]. The app is downloaded on the phone in which the data of user are inserted manually. The app neither shares the location of the user nor shares the user's identity. The technology disguises the user's identity by generating a random sequence of numbers that change after every few minutes. Using Bluetooth, the user's phone detects any other phone in proximity, which also has opted for the app. Both phones exchange these random numbers. The user, if tested positive for COVID-19, updates this information in the app. Every phone that was in contact with the COVID-19 patient's phone in last 14 days get notification that they were in contact with COVID-19 patient and should quarantine themselves. Singapore has made it compulsory for all the arriving passengers to wear monitoring wristbands to contain the spread of COVID-19. The wristband uses Global Positioning System (GPS) and Bluetooth to help the authorities in identifying the location of the user. The authorities will be able to ensure that the arriving passengers follow the quarantine rules defined by the government [32].

Clinical IoHT includes the monitoring of person under the supervision of physician as shown in Figure 2. A list of IoT-enabled clinical applications is presented in [33]. The list

includes devices to monitor the spread of cancer, continuous glucose monitoring, connected inhalers, asthma monitor, and many more. During this global pandemic, many healthcare gadgets can provide opportunities of real-time remote supervision. These gadgets are smart enough to provide results, which can be seen remotely by medical staff [34]. There are limitations to these clinical IoHT devices as well, which may cause unreliable results [35]. It can have a system to activate alarm if any unforeseen situation occurs. In the perspective of COVID-19, IoT-based ventilators and temperature monitors can help in providing the timely assistance to patients. The patient status can be monitored remotely if ventilators are connected to the cloud. The IoT-based temperature monitoring device can help in keeping the real-time record of everyone in the database. The record can be checked in later date if required [36].

### 3. Challenges of IoT in the Wake of COVID-19

Implementing IoT is never an easy task to perform. Furthermore, when implementing IoT for COVID-19, there are many challenges involved, few of which are described below.

**3.1. Scalability.** With the advent of digital technology, the number of IoT devices is growing exponentially [37]. The reason is that they are not limited to only one application, but there are many applications of IoT, which are in practice these days. According to a recent survey, there is a massive

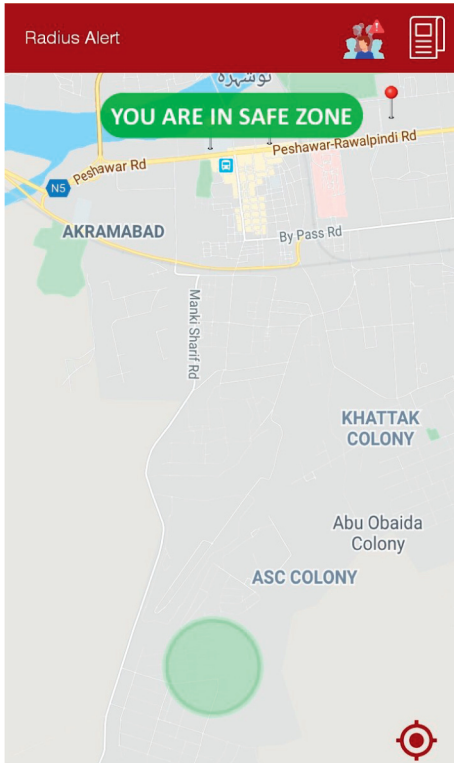


FIGURE 4: Real-time area monitoring for COVID-19 patients.

increase in the use of home automation appliances from 2018 to 2022 [38]. The trend is represented in Table 1.

Scalability is a big challenge in implementing IoT to fight against the global pandemic of COVID-19. A large number of devices are required in IoT alone to accurately sense the vital signs of the patients and forward those to the Internet cloud. As for now, the active cases are approximately 3.7 million worldwide. Each IoT gadget needs to have multiple sensors. Implementing IoT for this highly scaled scenario is a big challenge. The devices required are large in number, and large amount of data will float around these small IoT nodes.

Forecasts show that the IoT connections will double from 2019 to 2024 by reaching 24 billion [39]. Currently, there are no cases reported in which the scalability was an issue to interrupt seamless transmission of IoT devices, but the reports show that the Internet usage has been increased to 60% during the pandemic [40]. The IoT devices in health care have not yet been groomed to its full potential, but the demand will increase with time [41]. Due to scalability, the energy requirements and the need of accurate real-time performance in noisy environment have also increased [42]. With the sheer volume of data generated by a large number of distributed sensors, another challenge is to capture, integrate, store, and process the data [43]. Most of the healthcare IoT systems use rechargeable batteries, which have a very short life span. One potential approach to prolong the lifetime of the battery is to harvest energy [44].

**3.2. Security and Privacy Issues.** Due to the scalability and energy limitations of IoT devices, the security solutions should be energy-efficient and algorithms defined to secure the IoT

TABLE 1: Expected growth in domestic IoT applications from 2018 to 2022 (in million units).

System	2018	2022	Growth (%)
Video entertainment	310.5 m	457.5 m	10
Home monitoring and security	97.7 m	244.9 m	26
Smart speakers	99.8 m	230.5 m	23
Lighting	37.7 m	104.6 m	29
Thermostats	13.6 m	37.5 m	29
Others	84.5 m	189.3 m	22

network should have less computational complexities to offer end-to-end data protection, consumer privacy, and secure authentication [45, 46]. Thus, lightweight security algorithms need to be designed in order to implement security in IoT. With the outbreak of coronavirus, the security requirements of IoT-enabled networks have increased. The security should be enabled for both three-layered and service-oriented architectures of IoT [47]. The security concerns in implementing IoT with respect to COVID-19 are as follows:

- (i) The data that are sent from the sensors attached to the body of COVID-19 patient should be accurate
- (ii) The data should successfully reach the destination
- (iii) The data should not be forged
- (iv) The data should not be intercepted from the communication path
- (v) The data stored in the memory of the IoT device should not be accessible to everyone

The security primitives should be taken considering IoT devices, which have low computational capabilities. Besides being lightweight, the required security algorithms should be accurate and must be able to keep user's trust intact [9]. Security primitives include specific attack detection, channel state masking, intrusion detection, localization, and data provenance. A single change to the data can cause major problems. For example, if any misleading change in medical health reports of COVID-19 patient generated by IoT devices is sent to doctors, then it can cause major problems during the global pandemic of COVID-19. Ensuring the trust of both IoT healthcare device user and medical staff receiving the reports from the remote devices is a big challenge.

With the advancement in AI and ML, many new doors have opened up for the researchers to contribute to reducing the spread of COVID-19. The use of these technologies is specifically promising in contact tracing. Besides all the advantages, the advent of these technologies has raised ethical challenges. The surveillance of social contact has caused the fear of disrupting the commitment of privacy and autonomy of social groups [48]. One of the biggest challenges is to provide solutions to the community without compromising the privacy and user's trust.

**3.3. Limited Spectrum and Bandwidth.** As the number of IoT devices is increasing, more bandwidth is required to send all the information from sensors to the cloud. At present, most



of the IoT devices use the licensed spectrum offered by the mobile operators. With the growth of these devices, the bandwidth requirements have also increased. The data face latency in Internet protocol (IP) networks, which sometimes cause erroneous data transfer because the data packet is retransmitted if it fails to reach the destination initially. So, latency might occur due to the number of retries establishing a connection and transmitting the message, due to the nature of the protocol to reliably transmit data. Currently, many IoT devices use 4G/LTE networks to perform their tasks. This limited spectrum of 3 G/LTE/4G will soon be not enough for large number of IoT devices [49].

During the pandemic of COVID-19, timely transfer of data from IoT devices to the concerning body is of utmost importance. Errors or delay in data may cause loss of precious human lives. If the bandwidth is high, the problems of latency and low data rates can be overcome.

#### 4. Strength, Weakness, Opportunity, and Threat Analysis

Strength, Weakness, Opportunity, and Threat (SWOT) analysis for IoT is shown in Table 2. The internal factors are comprised of strengths and weaknesses, which are limited to the organizations or researchers who want to implement IoT. The internal factors can be changed with time. Opportunities and threats are considered as external factors, which depend on the market and cannot be changed [50].

*4.1. Strengths.* Considering COVID-19 as test case, the accuracy of data in IoT is one of the strengths in implementing it. The sensors take real-time data from the environment and send it to the cloud [51]. This results in helping the patients to get on-time treatment, which can save many lives. If anyone has symptoms of COVID-19 and needs to consult physician, then IoT helps in providing platform of telehealth in which a person can take the advice of physician without visiting the hospital or clinic. This refers to the timely diagnosis of COVID-19. IoT can help in spreading the awareness related to the information and safety measures to take preemptive measures against coronavirus. Due to the importance of IoT to combat against the current global crisis of coronavirus, there is a high demand of IoT-based systems [52]. Integrating AI with IoT can help in better forecasting future needs to fight against COVID-19.

*4.2. Weaknesses.* The shortcomings and weaknesses cannot be ignored while considering the implementation of IoT to combat against this virus. Due to the requirements of large number of IoT devices and scalability, the data processing units should have high processing power. The data centers should be more to keep record of patients and related information. The whole IoT network should be highly secured, and the security algorithms should be designed in such a way that complexity is kept as low as possible [53]. As many devices will be sending data frequently to the cloud, the requirements of high bandwidth cannot be ignored. The mechanism should be designed where limited spectrum

should be efficiently used. This can be done by frequency planning and reuse mechanism.

*4.3. Opportunities.* The opportunities are huge by implementing IoT to combat this global crisis of COVID-19. With the increase in IoT applications, the IoT industry can help in providing the jobs in local markets and effectively take its part in boosting the economy of any country. The use of millimeter wave (mmWave)-based 5G has not yet come into play for IoT networks, which provides large bandwidth and high data rate. The implementation of IoT can bend the tech giants toward the use of this large bandwidth mmWave, which operates between 3 GHz and 300 GHz [54]. This will open new doors in many areas of wireless communication networks. Currently, software-defined radios, cognitive radio networks, and cooperative communication can be applied in existing IoT networks to efficiently use the spectrum by sensing the empty spaces in licensed bands and using them for its operations.

*4.4. Threats.* The threats as external factors are few comparatively. Currently, IoT devices are compatible with the manufacturer of the same vendor [55]. There is a dire need of compatibility to develop competition among the vendors by integrating the platforms. This will help in the integration of different applications and services, which will increase the quality of IoT operations, and the applications will evolve with time. Besides, the range of unlicensed bands is very less. Most of the communication in IoT either uses cellular network or 2.4 GHz of Industrial, Scientific, and Medical (ISM) frequency band, which may cause interference if proper planning is not performed [56].

#### 5. Solution to the Challenges in Combating COVID-19

The challenges involved in implementing the scalable IoT networks are undeniable, but solutions to these challenges are present in the literature, which can help in successfully deploying the IoT networks [9, 57]. Some of the prominent solutions are presented in the following subsections.

*5.1. Lightweight Security Algorithms for Scalable IoT.* Due to scalability, most of the IoT devices to measure vital signs are small in size and easily accessible. Measures must be taken to ensure that the data are protected and are efficiently received at the destination. In most cases, most of these healthcare IoT devices are not physically protected, so data security and provenance serve as the backbone for implementing IoT networks. Data can be easily forged if the proper security primitives are not used.

Various metrics such as angle of arrival, time of arrival, phasor information, and received signal strength indicators (RSSIs) can be used to develop lightweight security algorithms for IoT devices. Wireless channel characteristics of IoT healthcare devices are used to design the algorithms, which (i) protect the IoT devices and (ii) due to their low

TABLE 2: SWOT analysis of IoT in the perspective of global pandemic.

Internal factors	
Strengths	Weaknesses
Accuracy of data	High processing server/fusion centers are required
On-time treatment	Scalability of IoT devices
Timely diagnosis	Huge data centers and data aggregation
Information of safety measure	Security and privacy preservation
High demand of IoT-based systems	High bandwidth requirements
Accurate forecasting	Limited spectral resources
External factors	
Opportunities	Threats
Creation of awareness about the requirement of IoT	Compatibility of devices
Creation of jobs	Use of unlicensed bands
Toward 5G for higher bandwidths	
Software-defined radios	
Cooperative communication	

computational complexities work efficiently in scalable environments. Figure 5(a) shows the RSSI variations of connected IoT devices, while Figure 5(b) provides a better insight by applying the Savitzky–Golay filter to the results achieved in Figure 5(a). In case of no adversary in the IoT network, there is a linear relationship between the RSSI variations. The RSSI values are converted into binary streams by quantification. The binary stream is referred to as the link fingerprints. These link fingerprints are encoded with symmetric key, and the resultant is sent to the server where it computes the Pearson correlation coefficient using the link fingerprints of connected IoT devices.

The computation of the Pearson correlation coefficient (PCC) is a very simple technique yet very accurate to detect any adversary in the IoT network [9]. The PCC is computed for both attack and non-attack scenarios, which are achieved as 0.9762 and 0.0632, respectively. Figure 5(c) presents the scenario when the IoHT is under attack and the communication path is changed. The communication between IoHT devices 1 and 2 is via an adversary. The RSSI variations are not linear, and hence, low PCC is computed. Applying the designed algorithms, the energy consumption of these IoT healthcare devices is as low as 26.99 mJ for 128 bit key size. This helps in prolonging the battery life of small IoT devices because the energy dissipation is very less. Some of the notable lightweight security algorithms present in the literature are summarized in Table 3.

*5.1.1. Blockchain for Connected Healthcare Units and Privacy Preservation.* Blockchain is the rapidly growing technology, which became famous because of a virtual currency called Bitcoin. The use of blockchain is expanded to many fields [61]. Blockchain enables privacy and security for data sharing [62, 63]. A blockchain-based IoT system presented in [64] stores the private key at IoT device, while the public keys are stored at Ethereum. Blockchain can be implemented for connected healthcare units as shown in Figure 6 in which all healthcare units are connected to each other. Each healthcare unit acts as a block, and accurate data transfer is made possible by implementing blockchain-based IoT network.

For example, the medical record of a patient received from one healthcare unit to another can be verified by generating a HASH and comparing it with all the HASH values present in the ledger.

Blockchain technology can also be used to secure end-to-end data. The security and privacy preservation are made sure if IoT is integrated with blockchain. Important data of medical records and the record of all available healthcare kits and other resources are verified by any official by checking whether the record is in its authentic form or is it forged. The SHA2 algorithm is applied on the data along with the private key ( $K$ ) associated with medical healthcare unit, nonce ( $N$ ), and previous hash ( $HASH_p$ ). Mathematically,

$$HASH = SHA1(Data, K_i, N_i, HASH_p). \quad (1)$$

A difficulty level ( $d$ ) is selected based on which the miners mine the data and add it to the blockchain. In 1,  $N$  is iterated until the HASH is achieved according to the  $d$ . Figure 7 shows that as the payload size increases, the mining time also increases. For a 10 kB data, if the  $d = 3$ , the achieved mining time of a single miner is less than 0.2 seconds. The trade-off is between the data size and  $d$ , i.e., if the data size is high,  $d$  should be low. All HASH values are sent to the cloud where a distributed ledger is created. The data can be checked in later date for its authentication. Even a single-digit change in the report generates a different hash. Due to this, any forgery can be detected.

This can be applied to the supply chain in which each supply point becomes a block and adds its hash to a decentralized ledger. The data (which could be the count of equipment) can be verified at any stage or precisely at the destination by looking at the HASH values in the ledger. If all HASH values match, then the supply has reached successfully. Figure 8 shows the same procedure in which the ledger is updated with the HASH values generated by each block. At each block, the HASH is verifiable, while in the last block, the data are slightly changed and a different HASH is generated. While at the ledger, the HASH values generated and presented at the ledger are not matched with the HASH of block number 3.

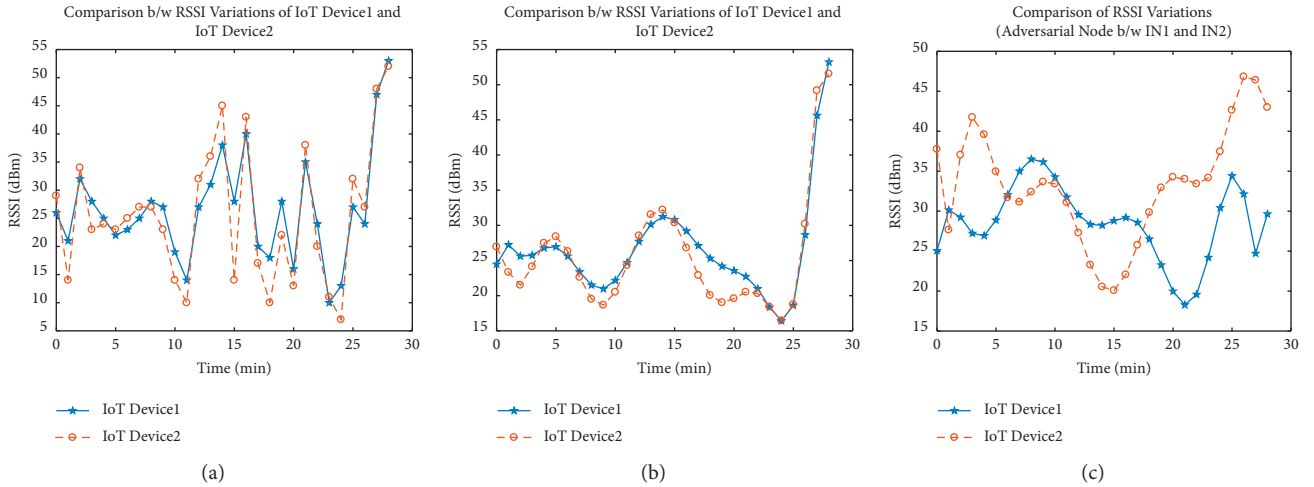


FIGURE 5: Comparison of RSSI values achieved from connected IoT devices in both attack and non-attack cases. (a) RSSI variations of connected IoT devices. (b) Applying the Savitzky–Golay filter to RSSI variations in (a). (c) Adversary in between IoT nodes.

TABLE 3: Lightweight security algorithms in the literature to combat various attacks in IoT network.

Security requirements	Gope and Sikdar [58]	Dong et al. [59]	Ali et al. [60]	Kamal and Tariq [9]
MITM attack	✓	✓	✓	✓
Jamming	✓	×	✓	✓
Data tempering	×	×	✓	✓
Replay attack	✓	×	×	×
Location proximity	×	×	×	✓
Data provenance	×	×	✓	✓

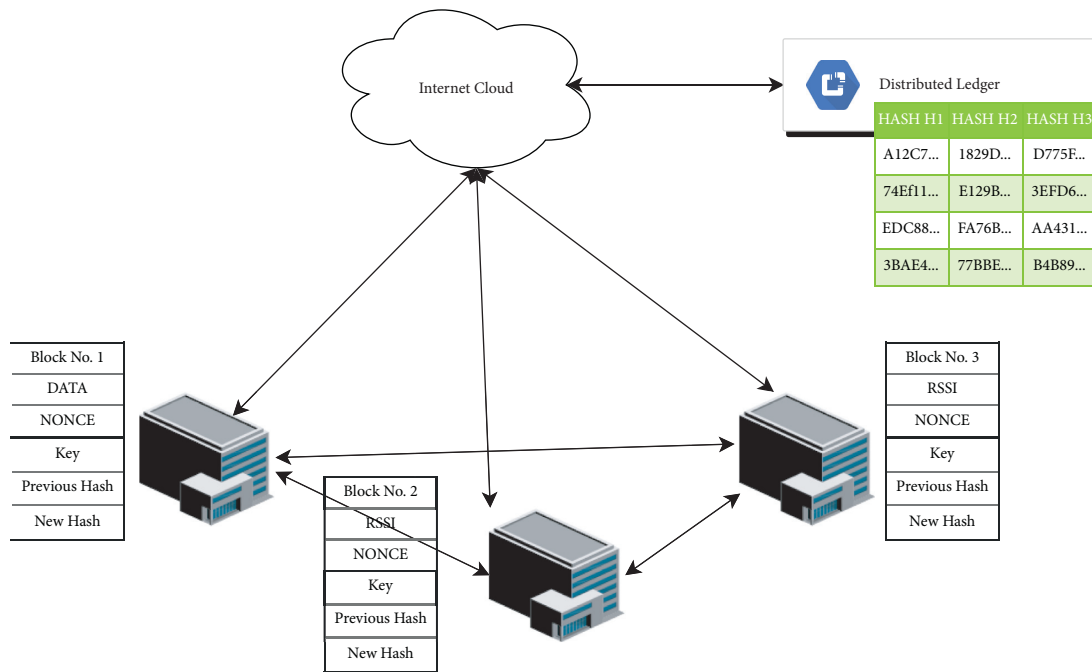


FIGURE 6: Blockchain-based connected healthcare units.

5.2. *Toward 5G for Higher Bandwidth.* With the advent of IoT, the demand of bandwidth has increased. For the organizations working on deploying IoT devices, the

bandwidth shortage has motivated them in the exploration of the underutilized mmWave frequency spectrum for future IoT networks. mmWave ranges from 3 GHz to 300 GHz [54].

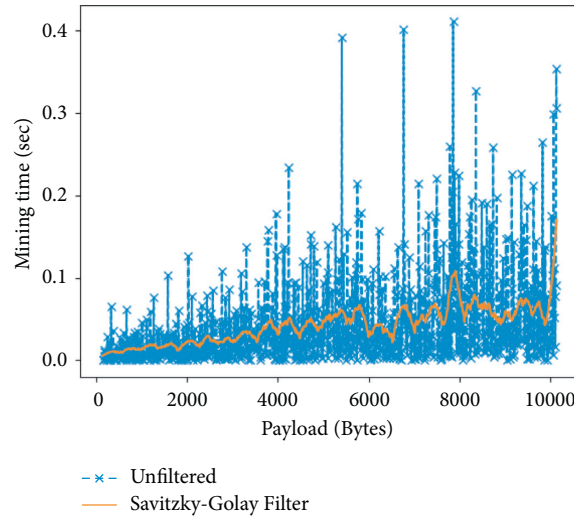


FIGURE 7: Blockchain-based connected healthcare units.

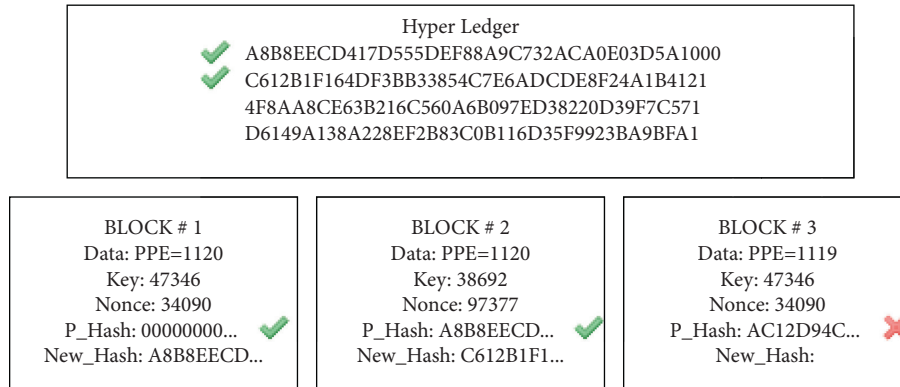


FIGURE 8: Blockchain-based security showing mismatch of hash in ledger for block number 3.

Spectrum at 28 GHz, 38 GHz, and 70–80 GHz looks especially promising for next-generation cellular systems. Because of large bandwidths, multi-gigabits per second can be achieved. 5G using mmWave spectrum provides promising benefits in other application scenarios such as wearable networks, vehicular communications, or autonomous robots [65].

As the frequency spectrum range is broad, more bandwidth is available at these frequencies. The capacity (C) is increased, which solves the problem of scalability in IoT networks because mathematically,

$$C = BW \times \log_2(1 + SNR), \quad (2)$$

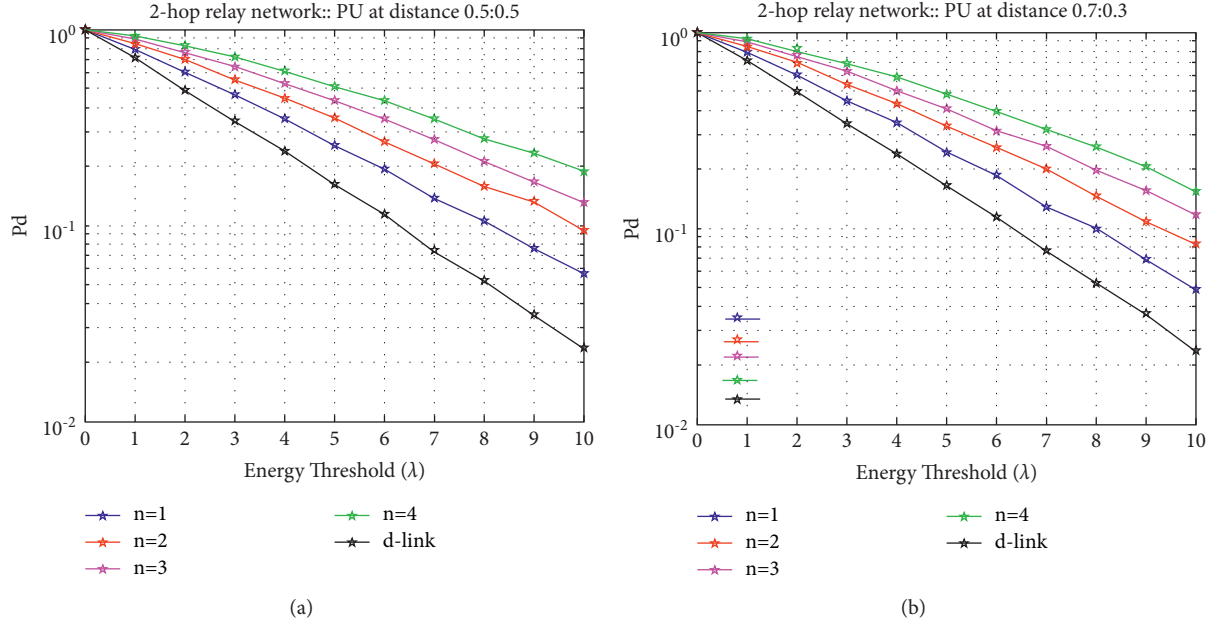
where BW represents the bandwidth and SNR is the signal-to-noise ratio. Due to higher attenuation in free space, the same frequency is reused at shorter distances. The security and privacy are better because of the limited range and narrow beamwidths [66]. As the frequency is high, then the wavelength is small, and hence, small antenna size helps in integrating the large array of antennas on a chip or printed circuit boards (PCBs). The comparison [67] of 3G and 4G (existing solutions for IoT

to send sensor data to cloud) with the 5G is presented in Table 4. It is evident that moving to 5G will help in the better performance of IoT devices [68], specifically in IoHT, where the low latency and high data rate are key factors for efficient and effective performance.

*5.2.1. Cognitive Radio-Enabled IoT.* Cognitive radio merged with IoT is called cognitive radio IoT (CRIoT) [57]. Spectrum allocation is always been done traditionally in a licensed fashion. It has been observed that most of the licensed spectrum is not completely utilized. Cognitive radios are proposed as a viable solution to the frequency reuse problem [69]. While using cognitive radio parameters, IoT devices are capable of sensing the environment and adjusting the configuration parameters automatically [70]. The IoT devices sense the availability of free spectrum referred as holes in the spectrum and communicate in the sensed holes without interfering with the licensed user called primary user (PU) [71]. This helps in uninterrupted data communication and efficient utilization of licensed spectrum.

TABLE 4: Comparison of 3G, 4G, and 5G technologies.

Parameter	3G	4G	5G
Frequency range	1.8 – 2.5 GHz	2 – 8 GHz	3 – 300 GHz
Latency	100 – 500 ms	20 – 30 ms	<10 ms
Data rate	2 Mbps	2 Mbps – 1 Gbps	>1 Gbps

FIGURE 9:  $P_d$  for the various cases based on the position of relay in CRIoT. (a) Relay is equidistant from PU and SU. (b) Relay is placed close to SU and far from PU.

This can be achieved by deploying relays in the CRIoT. The relays sense the spectrum and provide the unused bands to the secondary users (SUs). The real challenge arises in the deployment of these relays, i.e., where to place these relays. The simulations are performed for various test cases in which energy of the spectrum is sensed against a defined ( $T_d$ ). The results are presented in Figure 9 in which it is observed that as the relays are placed closer to the SU and away from PU (Figure 9(b)), the probability of detection ( $P_d$ ) of sensing the PU in the CRIoT decreases.

## 6. Conclusion

During the outbreak of the global challenge of COVID-19 pandemic, the reliance on technologies such as IoT, AI, blockchain, big data analytics, and cloud computing has increased. IoT plays a major role in reducing the risks of coronavirus spread by providing platforms, which help in following the protocols defined by WHO. IoT-based healthcare units provide timely response by medical staff to deal with COVID-19 patients. Blockchain-based IoT networks help in better management of the supply chain and detect any forgery in data. The challenges in implementing IoT networks cannot be ignored. To deal with the scalability, 5G using mmWave-based communication system provides support to enable end-to-end communication. The need for

lightweight security is also obvious because the IoT devices are small in size and large in number. The solution suggests to implement algorithms, which have less computational cost.

## Data Availability

The data are available upon request to Dr. Mohsin Kamal (kamal.mohsin@ucy.ac.cy).

## Conflicts of Interest

The authors declare that they have no conflicts of interest to report regarding this study.

## Acknowledgments

The authors would like to acknowledge the support from Taif University Researchers Supporting Project Number (TURSP-2020/331), Taif University, Taif, Saudi Arabia. This work was supported by the Ministry of Health (MoH), Saudi Arabia, Grant No. 935. The authors, therefore, gratefully acknowledge the MoH technical and financial supports. Furthermore, the previous version of this manuscript is present as arXiv at Cornell University's website (<https://arxiv.org/abs/2007.12268>) [72].

## References

- [1] C. R. Srinivasan, B. Rajesh, P. Saikalyan, K. Premsagar, and E. S. Yadav, "A review on the different types of Internet of Things (IoT)," *Journal of Advanced Research in Dynamical and Control Systems*, vol. 11, no. 1, pp. 154–158, 2019.
- [2] A. Panarello, N. Tapas, G. Merlino, F. Longo, and A. Puliafito, "Blockchain and IoT integration: a systematic survey," *Sensors*, vol. 18, no. 8, p. 2575, 2018.
- [3] M. A. A. Garadi, A. Mohamed, A. K. A. Ali, X. Du, I. Ali, and M. Guizani, "A survey of machine and deep learning methods for internet of things (iot) security," *IEEE Communications Surveys & Tutorials*, vol. 22, 2020.
- [4] K. D. Saranya, R. Krishnamurthy, K. N. H. Srinivas, T. S Rao, and I. S. Amiri, "IoT-based health monitoring system using beaglebone black with optical sensor," *Journal of Optical Communications*, vol. 1, 2019.
- [5] D. Minovski, C. Åhlund, and K. Mitra, "Modeling quality of IoT experience in autonomous vehicles," *IEEE Internet of Things Journal*, vol. 7, no. 5, pp. 3833–3849, 2020.
- [6] S. Aheleroff, X. Xu, Y. Lu et al., "IoT-enabled smart appliances under industry 4.0: a case study," *Advanced Engineering Informatics*, vol. 43, Article ID 101043, 2020.
- [7] M. Kamal, G. Srivastava, and M. Tariq, "Blockchain-based lightweight and secured V2V communication in the internet of vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 7, pp. 3997–4004, 2021.
- [8] X. C. Yin, Z. G. Liu, L. Nkenyereye, and B. Ndiabanje, "Toward an applied cyber security solution in IoT-based smart grids: an intrusion detection system approach," *Sensors*, vol. 19, no. 22, p. 4952, 2019.
- [9] M. Kamal and M. Tariq, "Light-weight security and data provenance for multi-hop Internet of Things," *IEEE Access*, vol. 6, Article ID 34439, 2018.
- [10] Ministry National Health Services, "COVID-19 Global," 2021, <http://covid.gov.pk/stats/global/>.
- [11] S. Mavrikou, G. Moschopoulou, V. Tsekouras, and S. Kintzios, "Development of a portable, ultra-rapid and ultra-sensitive cell-based biosensor for the direct detection of the SARS-CoV-2 S1 spike protein antigen," *Sensors*, vol. 20, no. 11, 2020.
- [12] Z. Allam and D. S. Jones, "On the coronavirus (COVID-19) outbreak and the smart city network: universal data sharing standards coupled with artificial intelligence (AI) to benefit urban health monitoring and management," *Healthcare*, vol. 8, no. 1, p. 46, 2020.
- [13] F. Hussain, R. Hussain, S. A. Hassan, and E. Hossain, "Machine learning in iot security: current solutions and future challenges," *IEEE Communications Surveys & Tutorials*, vol. 22, 2020.
- [14] N. Saeed, B. Ahmed, T. Y. A. Naffouri, and M.-S. Alouini, "When Wireless Communication Faces COVID-19: Combating the Pandemic and Saving the Economy," 2020, <https://arxiv.org/abs/2005.06637>.
- [15] J. J. P. C. Rodrigues, D. B. D. R. Segundo, H. A. Junqueira et al., "Enabling technologies for the Internet of health things," *IEEE Access*, vol. 6, Article ID 13129, 2018.
- [16] J. E. Hollander and B. G. Carr, "Virtually perfect? Telemedicine for COVID-19," *New England Journal of Medicine*, vol. 382, no. 18, pp. 1679–1681, 2020.
- [17] A. Poppas, J. S. Rumsfeld, and J. D. Wessler, "Telehealth is having a moment: will it last?" *Journal of the American College of Cardiology*, vol. 75, 2020.
- [18] A. C. Smith, E. Thomas, C. L. Snoswell et al., "Telehealth for global emergencies: implications for coronavirus disease 2019 (COVID-19)," *Journal of Telemedicine and Telecare*, vol. 26, Article ID 1357633X20916567, 2020.
- [19] B. Siwicki, "Updated: a guide to connected health device and remote patient monitoring vendors," 2020, <https://www.healthcareitnews.com/news/guide-connected-health-device-and-remote-patient-monitoring-vendors/>.
- [20] Health Arc, "Remote patient monitoring made easy," 2019, <https://web.healtharc.io/>.
- [21] Continuous Care, "Better health outcomes guaranteed," 2021, <https://www.continuouscare.io/remote-monitoring/>.
- [22] HealthnetConnect, "Healthcare Delivery, remimagined," 2021, <https://healthnetconnect.com/>.
- [23] Sehatyab, "Tele-medicine to resuscitate primary care in pakistan," 2021, <https://sehatyab.com/hazarnaimat/tele-medicine-resuscitate-primary-care-pakistan/>.
- [24] E. Park, J. H. Kim, H. S. Nam, and H. J. Chang, "Requirement analysis and implementation of smart emergency medical services," *IEEE Access*, vol. 6, Article ID 42022, 2018.
- [25] Was, "Making vehicles special," <https://www.was-vehicles.com/en/home.html>.
- [26] H. Habibzadeh, K. Dinesh, O. R. Shishvan, A. B. Dandry, G. Sharma, and T. Soyata, "A survey of healthcare internet-of-things (hiot): a clinical perspective," *IEEE Internet of Things Journal*, vol. 7, 2019.
- [27] M. Ghanavatinejad, M. Tavakoli, and M. M. Sepehri, "A clustering model for gadgets and apps in patient monitoring in HIOT environment in health field," *Journal of Hospital*, vol. 18, no. 1, pp. 21–30, 2019.
- [28] Apple Inc, "The No.1 smartwatch in the world. Times two," 2021, <https://www.apple.com/lae/watch/>.
- [29] Fitbit Inc, "At fitbit, health & fitness come first," 2021, <https://www.fitbit.com/global/eu/home>.
- [30] Apple Inc, "Apple and google partner on covid-19 contact tracing technology," 2020, <https://www.apple.com/newsroom/2020/04/apple-and-google-partner-on-covid-19-contact-tracing-technology/>.
- [31] Google Inc, "Exposure notifications: using technology to help public health authorities fight covid-19," 2020, <https://www.google.com/covid19/exposurenotifications/>.
- [32] AS inc, "Singapore to make arriving passengers wear monitoring wristbands," 2020, [https://en.as.com/en/2020/08/06/latest\\_news/1596723358\\_452783.html/](https://en.as.com/en/2020/08/06/latest_news/1596723358_452783.html/).
- [33] Econsultancy, "10 Examples of the internet of things in healthcare," 2019, <https://econsultancy.com/internet-of-things-healthcare/>.
- [34] H. H. Nguyen, F. Mirza, M. A. Naeem, and M. Nguyen, "A review on iot healthcare monitoring applications and a vision for transforming sensor data into real-time clinical feedback," in *Proceedings of the 2017 IEEE 21st International Conference on Computer Supported Cooperative Work in Design (CSCWD)*, pp. 257–262, IEEE, Wellington, New Zealand, 2017.
- [35] I. U. Din, A. Ahmad, M. Guizani, and M. Zuair, "A decade of internet of things: analysis in the light of healthcare applications," *IEEE Access*, vol. 7, Article ID 89967, 2019.
- [36] M. N. Mohammed, H. Syamsudin, S. Al-Zubaidi, R. A. K. S. Ramli, and E. Yusuf, "Novel COVID-19 detection and diagnosis system using IOT based smart helmet," *International Journal of Psychosocial Rehabilitation*, vol. 24, no. 7, 2020.
- [37] A. Gupta, R. Christie, and P. R. Manjula, "Scalability in internet of things: features, techniques and research challenges," *International Journal of Computational Intelligence Research*, vol. 13, no. 7, pp. 1617–1627, 2017.

- [38] F. Richter, "Infographic: smart home technology poised for blockbuster growth," 2018, <https://www.statista.com/chart/15736/smart-home-market-forecast/>.
- [39] G. S. M. Associations, "Iot connections forecast: the impact of covid-19," 2020, <https://www.gsma.com/iot/resources/iot-connections-forecast-the-impact-of-covid-19/>.
- [40] B. Ryan, "The internet is under huge strain because of the coronavirus. experts say it can cope — for now," 2020, <https://www.cnn.com/2020/03/27/coronavirus-can-the-internet-handle-unprecedented-surge-in-traffic.html/>.
- [41] Eseyes .inc., "Internet of healthcare things (ioht) trends," 2020, <https://www.eseye.com/internet-of-healthcare-things-ioht-trends/>.
- [42] E. Sisinni, A. Saifullah, S. Han, U. Jennehag, and M. Gidlund, "Industrial internet of things: challenges, opportunities, and directions," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 11, pp. 4724–4734, 2018.
- [43] H. Cai, B. Xu, L. Jiang, and A. V. Vasilakos, "IoT-based big data storage systems in cloud computing: perspectives and challenges," *IEEE Internet of Things Journal*, vol. 4, no. 1, pp. 75–87, 2016.
- [44] A. S. Adila, A. Husam, and G. . Husi, "Towards the self-powered internet of things (iot) by energy harvesting: trends and technologies for green iot," in *Proceedings of the 2018 2nd International Symposium on Small-scale Intelligent Manufacturing Systems (SIMS)*, 2018.
- [45] M. N. Aman, M. H. Basheer, and B. Sikdar, "Data provenance for iot with light weight authentication and privacy preservation," *IEEE Internet of Things Journal*, vol. 6, no. 6, Article ID 10441, 2019.
- [46] M. N. Aman, M. H. Basheer, and B. Sikdar, "A lightweight protocol for secure data provenance in the internet of things using wireless fingerprints," *IEEE Systems Journal*, vol. 15, 2020.
- [47] J. Lin, W. Yu, N. Zhang, X. Yang, H. Zhang, and W. Zhao, "A survey on internet of things: architecture, enabling technologies, security and privacy, and applications," *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1125–1142, 2017.
- [48] D. Leslie, "Tackling COVID-19 through responsible AI innovation: five steps in the right direction," *Harvard Data Science Review*, 2020.
- [49] M. Cudak, A. Ghosh, T. Kovarik et al., "Moving towards mmwave-based beyond-4G (B-4G) technology," in *Proceedings of the 2013 IEEE 77th Vehicular Technology Conference (VTC Spring)*, 2013.
- [50] T. Berry, "How to Do a SWOT Analysis for Better Strategic Planning," 2021, <https://articles.bplans.com/how-to-perform-swot-analysis/>.
- [51] S. B. Baker, W. Xiang, and I. Atkinson, "Internet of things for smart healthcare: technologies, challenges, and opportunities," *IEEE Access*, vol. 5, Article ID 26521, 2017.
- [52] M. S. Rahman, N. C. Peeri, N. Shrestha, R. Zaki, U. Haque, and S. H. A. Hamid, "Defending against the novel coronavirus (covid-19) outbreak: how can the internet of things (iot) help to save the world?" *Health Policy and Technology*, vol. 9, 2020.
- [53] L. D. Xu, W. He and S. Li, "Internet of things in industries: a survey," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 4, pp. 2233–2243, 2014.
- [54] T. S. Rappaport, S. Sun, R. Mayzus et al., "Millimeter wave mobile communications for 5G cellular: it will work!" *IEEE Access*, vol. 1, pp. 335–349, 2013.
- [55] S. Sonune, D. Kalbande, A. Yeole, and S. Oak, "Issues in IoT healthcare platforms: a critical study and review," in *Proceedings of the 2017 International Conference on Intelligent Computing and Control (I2C2)*, 2017.
- [56] T. S. Rappaport, "Wireless communications—principles and practice, (the book end)," *Microwave Journal*, vol. 45, no. 12, pp. 128–129, 2002.
- [57] D. Tarek, A. Benslimane, M. Darwish, and A. M. Kotb, "A New Strategy for Packets Scheduling in Cognitive Radio Internet of Things," vol. 178, *Computer Networks*, Article ID 107292, 2020.
- [58] P. Gope and B. Sikdar, "Lightweight and privacy-preserving two-factor authentication scheme for IoT devices," *IEEE Internet of Things Journal*, vol. 6, no. 1, pp. 580–589, 2019.
- [59] Z. Dong, R. Espejo, Y. Wan, and W. Zhuang, "Detecting and locating man-in-the-middle attacks in fixed wireless networks," *Journal of Computing and Information Technology*, vol. 23, no. 4, pp. 283–293, 2015.
- [60] S. T. Ali, V. Sivaraman, D. Ostry, G. Tsudik, and S. Jha, "Securing first-hop data provenance for bodyworn devices using wireless link fingerprints," *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 12, pp. 2193–2204, 2014.
- [61] V. Ribeiro, R. Holanda, A. Ramos, and J. J. P. C. Rodrigues, "Enhancing key management in lorawan with permissioned blockchain," *Sensors*, vol. 20, no. 11, 2020.
- [62] A. Celesti, A. Ruggeri, M. Fazio, A. Galletta, M. Villari, and A. Romano, "Blockchain-based healthcare workflow for tele-medical laboratory in federated hospital IoT clouds," *Sensors*, vol. 20, no. 9, p. 2590, 2020.
- [63] A. Ahmed, Q. Nasir, and M. A. Talib, "Blockchain for government services—use cases, security benefits and challenges," in *Proceedings of the Fifteenth Learning and Technology Conference (L&T)*, p. 112, 2018.
- [64] S. Huh, S. Cho, and S. Kim, "Managing IoT devices using blockchain platform," in *Proceedings of the Nineteenth International Conference on Advanced Communication Technology (ICACT)*, pp. 464–467, IEEE, PyeongChang, Republic of Korea, 2017.
- [65] T. Lv, Y. Ma, J. Zeng, and P. T. Mathiopoulos, "Millimeter-wave NOMA transmission in cellular M2M communications for internet of things," *IEEE Internet of Things Journal*, vol. 5, no. 3, pp. 1989–2000, 2018.
- [66] T. Yilmaz and B. A. Ozgur, "On the use of the millimeter wave and low terahertz bands for internet of things," in *Proceedings of the 2015 IEEE second World Forum on Internet of Things (WF-IoT)*, pp. 177–180, IEEE, Milan, Italy, 2015.
- [67] E. Ezhilarasan and M. Dinakaran, "A review on mobile technologies: 3g, 4g and 5g," in *Proceedings of the Second International Conference on Recent Trends and Challenges in Computational Models (ICRTCCM)*, pp. 369–373, IEEE, Tindivanam, India, 2017.
- [68] D. Wang, D. Chen, B. Song, N. Guizani, X. Yu, and X. Du, "From iot to 5g i-iot: the next generation iot-based intelligent algorithms and 5g technologies," *IEEE Communications Magazine*, vol. 56, no. 10, 2018.
- [69] A. Shakeel, R. Hussain, A. Iqbal, I. L. Khan, Q. u. Hasan, and S. A. Malik, "Analysis of efficient spectrum handoff in a multi-class hybrid spectrum access cognitive radio network using Markov modelling," *Sensors*, vol. 19, no. 19, p. 4120, 2019.
- [70] J. A. Ansere, M. Kamal, E. Gyamfi, F. Sam, M. Tariq, and A. Mohammed, "Energy efficient resource optimization in cooperative Internet of Things networks," *Internet of Things*, vol. 12, Article ID 100302, 2020.
- [71] S. Mir, I. Bari, M. Kamal, and H. Ali, "Constraint waveform design for spectrum sharing under coexistence of radar and communication systems," *IEEE Access*, vol. 9, Article ID 46093, 2021.
- [72] M. Kamal, A. Aljohani, and E. Alanazi, "IoT meets covid-19: status, challenges, and opportunities," 2020, <https://arxiv.org/abs/2007.12268>.

## Research Article

# Image Splicing-Based Forgery Detection Using Discrete Wavelet Transform and Edge Weighted Local Binary Patterns

Muhammad Hameed Siddiqi <sup>1</sup>, Khurshed Asghar,<sup>2</sup> Umar Draz ,<sup>3,4</sup> Amjad Ali <sup>3</sup>,  
Madallah Alruwaili <sup>1</sup>, Yousef Alhwaiti <sup>1</sup>, Saad Alanazi <sup>1</sup>, and M. M. Kamruzzaman <sup>1</sup>

<sup>1</sup>College of Computer and Information Sciences, Jouf University, Sakaka, Al-Jouf 2014, Saudi Arabia

<sup>2</sup>Department of Computer Science, University of Okara, Okara, Pakistan

<sup>3</sup>Department of Computer Science, COMSATS University Islamabad, Lahore Campus, Islamabad, Pakistan

<sup>4</sup>Department of Computer Science, University of Sahiwal, Sahiwal, Pakistan

Correspondence should be addressed to Umar Draz; sheikhumar520@gmail.com

Received 25 June 2021; Accepted 8 September 2021; Published 30 September 2021

Academic Editor: Usman Habib

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

With the advancement of the multimedia technology, the extensive accessibility of image editing applications makes it easier to tamper the contents of digital images. Furthermore, the distribution of digital images over the open channel using information and communication technology (ICT) makes it more vulnerable to forgery. The vulnerabilities in telecommunication infrastructure open the doors for intruders to introduce deceiving changes in image data, which is hard to detect. The forged images can create severe social and legal troubles if altered with malicious purpose. Image forgery detection necessitates the development of sophisticated techniques that can efficiently detect the alterations in the digital image. Splicing forgery is commonly used to conceal the reality in images. Splicing introduces high contrast in the corners, smooth regions, and edges. We proposed a novel image forgery detection technique based on image splicing using Discrete Wavelet Transform and histograms of discriminative robust local binary patterns. First, a given color image is transformed in YCbCr color space and then Discrete Wavelet Transform (DWT) is applied on Cb and Cr components of the digital image. Texture variation in each subband of DWT is described using the dominant rotated local binary patterns (DRLBP). The DRLBP from each subband are concatenated to produce the final feature vector. Finally, a support vector machine is used to develop image forgery detection model. The performance and generalization of the proposed technique were evaluated on publicly available benchmark datasets. The proposed technique outperformed the state-of-the-art forgery detection techniques with 98.95% detection accuracy.

## 1. Introduction

Digital imaging is applicable in many fields such as World Wide Web (WWW), print media, insurance industry, and surveillance security [1]. All these applications leverage information and communication technology (ICT) to disseminate the digital contents including digital images [2]. The vulnerabilities in telecommunication infrastructure open the doors for intruders to access or change the transmitted data. The change in image data is hard to detect because contents of an image can be easily manipulated with the help of sophisticated image editing tools. The society is facing problems like false propaganda, fraud, counterfeiting, black

mailing, etc., due to image tampering. Image authentication is required to use images as source of information or evidence in real life. In most of the cases, especially with malicious designs, image forgery is performed using copy-move and splicing procedures. During forgery process, images can be altered with the help of the same image contents or by combining contents of different images. If the tampering procedure involves the copy and paste operation of image content/s within the image, then this forgery is called copy-move; otherwise, it is referred to as splicing. Different types of postprocessing operations such as scaling, blurring, noise adding, compression, and rotation are applied on the forged regions to hide the cues of forgery [3].



Forensic analysis of images was initiated in 2000. Many techniques [4–8] were developed to detect splicing forgery and can be categorized as active and passive (or blind) on the basis of splicing detection mechanism. Active techniques work on the phenomena that given image contains the information such as watermark or signature at the time of acquisition to ensure its authenticity. Active techniques extract this watermark or signature with the original to ensure its authenticity. The use of these techniques is very limited due to the nonavailability of information about the watermark or signature in most of the cases. Due to this limitation passive techniques for splicing forgery detection are being developed, which do not depend on prior information. In image splicing the contents of host images are modified by copying and pasting the contents from other images. Splicing is a fundamental and famous image forgery technique. To gain the public trust in digital image authentication, image splicing detection has become an important research area for digital image forgery detection. The image splicing operation disturbs the contents consistency, smoothness, and regularity. These factors play a very important role in detecting the forgery regions in the host image. The state-of-the-art image splicing techniques consider the variations in global statistical characteristics introduced by sudden inconsistency in spliced images [6–9]. Example of splicing image forgery is shown in Figure 1.

To detect the discontinuities that occurred in image due to splicing, first a given image is partitioned into subbands using DWT. The strong decorrelation ability of DWT represents the coefficients of four wavelet subbands at the same level. Our proposed technique measures the discontinuities that occurred in images due to splicing using the DWT subbands coefficients. The proposed scheme uses a robust technique for coding, which encodes the DWT subbands coefficients. Based on the proposed scheme which measures discontinuities and their coding, we introduce a new technique to detect splicing forgery by decomposing chroma components of a test image using DWT into subbands for measuring local discontinuities. For coding, we applied the DRLBP to determine the local discontinuities. We call the descriptor based on these methods as the DWT-DRLBP, which represents an image and is used for detecting splicing forgery as shown in Figure 2. Finally, the SVM is used to detect the image forgery in digital images.

## 2. Related Work

Most of the splicing forgery detection techniques are blind/passive [7]. Alahmadi et al. [9] and Min and Dong [10] used DCT coefficients and minimum and maximum filters to extract features from image blocks to detect splicing forgery. Multiresolution approaches, like DWT, are used in many algorithms [5, 11]. SIFT features are used as an alternative to block matching for detecting splicing forgery [12]. Most of the splicing forgery detection methods are evaluated on Columbia Color DVMM [13] and CASIA v1.0 and CASIA v2.0 [14] datasets. Ng et al. [15] proposed image splicing detection approach based on 3D moments of image spectrum, while features based on camera response

function were passed to SVM in [16] to detect splicing forgery. Shi et al. [17] used 1D and 2D moments, Markov chain probabilities, and DCT coefficients for image splicing detection. The algorithm was evaluated on CASIA v2.0 dataset and reported accuracy is 84.86%. Xunyu et al. [5] enhanced the accuracy to 89.76% of Wang et al. method by concatenating Markov chain moments and DCT and DWT coefficients together with SVM. Markov probabilities were extracted from Cb channel in [18], for image splicing detection. The algorithm achieved 89.23% and 95.5% accuracy, respectively, when evaluated on Columbia Color DVMM and a subset of CASIA v2.0 datasets. Zaho et al. in [19] designed a chrominance channel to detect splicing forgery and improved the performance of the Wang et al. scheme proposed in [18].

With the recent development in ubiquitous computing and digital media, especially digital images, the image forgery detection has become most essential task for secure and authentic multimedia contents transmission. Alahmadi et al. [20] used DCT and LBP features for image splicing detection. Pham et al. [21] extracted Markov features to identify irregularities in images due to splicing. SVM was used for classification. Jalab et al. [22] extracted fractional entropy from DWT [23] coefficients and SVM was used for classification. Xunyu in [5] developed an efficient technique to detect the duplicate regions from forged image. The proposed technique detects the *key points* using geometric transforms to find the identical transformed regions. Similarly, Mahmoud and Hongli in [6] developed a two-level key point detection technique to highlight the image tampering effects in smooth regions. At first level the combination of scale invariant feature operator and Harris corner detector was applied to detect the key point features from smooth regions. Finally, the gradient histogram of multisupport region order descriptor was computed to efficiently detect the tampers regions in a forged image.

Min and Dong in [10] developed a novel forgery detection technique based on minimum and maximum filter. The combination of minimum filter and maximum filter highlights the pixel wise minimum and maximum variations between authentic and forged images. The investigation of interpolation and noninterpolation improved the performance of forgery detection technique in composite regions. Recently, Jinwei et al. in [11] proposed a novel deep learning technique for image splicing detection. The proposed convolutional neural network learns the weighted combination of three types of featured extraction techniques. Convolutional neural network model learns the optimal combination of parameters for feature extraction techniques. Figure 3 demonstrates that sample image is transformed to YCbCr color space, where Cb and Cr are the chroma components and Y is the luminance component. Actually, the contents of image are described by luminance channel, which is capable of hiding the content tampering traces.

In [24], authors proposed a solution to localize image splicing using Multitask Fully Convolutional Network (MFCN). The proposed scheme achieves better performance than the single task FCN scheme. In the proposed scheme, authors used FCN VGG-16 with skip connection as the base



FIGURE 1: (a) Original image and (b) spliced image.

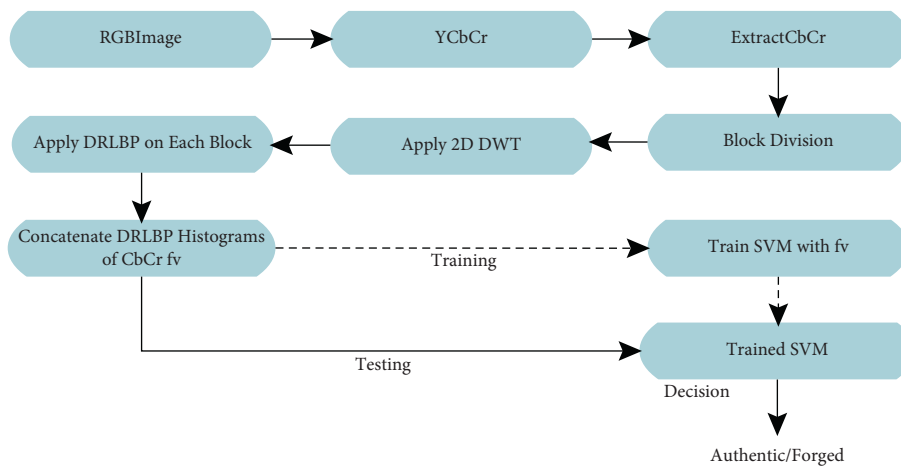


FIGURE 2: Proposed splicing image forgery detection approach.

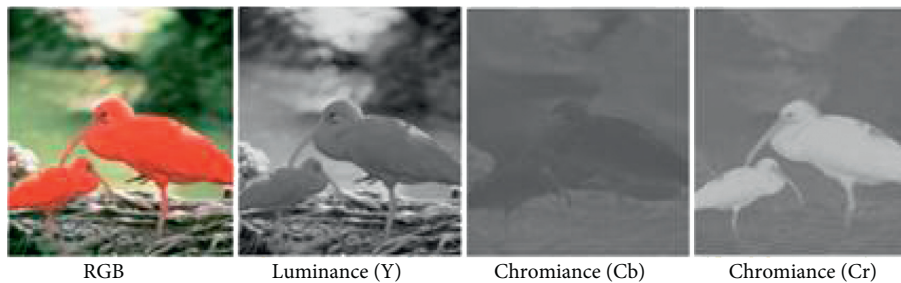


FIGURE 3: YCbCr components of an RGB image.

network, in order to improve the learning way of CNN through recall and consolidation mechanism of human brain. Bi et al. in [25] proposed a CNN-based method called Ringed Residual U-Net (RRU-Net). The proposed scheme is end-to-end image segmentation network for image splicing detection. In this scheme, residual propagation is used to recall the input feature information to solve the degradation problem in the deeper network. The RRU-Net was tested on CASIA and Columbia datasets which were also used by Wang et al. in [26] to detect and locate image forgeries. In order to detect copy-move forgery, a two-branch DNN

based architecture called BusterNet is proposed in [27]. In the proposed scheme, one layer is used for the detection of cloned regions which takes input image and uses CNN to extract features, self-correlation module to compute feature similarity, percentile pooling to collect useful statistics, mask decoder for upsampling of feature map, and binary classifier to generate binary copy-move mask, while the other layer is used for detection of tampered regions which takes input image, extracts features using CNN, upsamples feature map using mask decoder, and generates mask using binary classifier.

### 3. Proposed Image Forgery Detection Scheme Using the DWT-DRLBP Descriptor

In this section, we will discuss our proposed image forgery detection scheme using the DWT and the DRLBP descriptors. Splicing distorts the texture patterns that define the sharp changes such as corners, lines, and edges. Such inconsistencies in image texture can be easily detected with chroma components, because the chroma components describe the weak signals (corners, lines, and edges) [7, 18, 19]. Splicing highlights the discontinuities in the form of edges in images which change the local structure of spliced images and are well exposed using DWT coefficients, because the changes that occurred due to splicing are present in high frequency wavelet bands. To analyze these changes we propose an efficient, simple, and robust descriptor, called DWT-DRLBP descriptor, which first decomposes chroma components of a given image into subbands using Discrete Wavelet Transform (DWT) and then encodes these subbands using DRLBP [28] texture descriptor, which is a robust texture descriptor. The DWT-DRLBP descriptor of an image is passed to SVM for taking the decision whether it is authentic or spliced. We applied SVM two-class classifier to classify a sample image as forged or authentic [29]. Support vector machine is linear classifier, but the samples of image forgery dataset used in this research are not linearly separable. To overcome this issue a kernel trick is applied. The experiments were performed using LIBSVM kernel as presented in [30].

**3.1. DWT-DRLBP Descriptor.** Discrete wavelet transformation decomposed the sample image into four frequency bands (LL, LH, HL, and HH). These frequency bands are called chroma components, which highlights the local inconsistencies in the forged. The proposed image splicing technique is the combination of DWT based chroma components and DRLBP features. For this purpose, first one-level DWT was applied in the image and then DRLBP descriptor was applied to highlight the splicing effects.

**3.2. Wavelet Decomposition of Chroma Components.** The DWT provides unique and discriminatory representation to quantify image texture efficiently with high resolution and few numbers of wavelet coefficients. The wavelet coefficients effectively highlight the structural variations in image splicing. The low frequency coefficients provide high contrast that occurs due to image splicing. The low frequency features are directly used to represent the sample image. Due to describing energy compaction in few wavelet coefficients, the procedure of image representation becomes very simple.

The low frequency components image approximations are done by highlighting the inconsistencies introduced in the forged image. The low frequency is the most suitable for localization of variations in image contents as illustrated in Figure 4. The introduction of wavelet transformation in image splicing detection allows analyzing the image at frequency domain with the help of low-pass filter and high-pass filter. The splicing forgery produces high contrast in terms of corners, edges, and lines, which are better described

with high frequency. The wavelet transformation describes these transitions with the help of local sharpness and smoothness in high frequency coefficients. Based on these assumptions, each chroma channel of a given image is partitioned into four frequency bands (LL, LH, HL, and HH) using 1-level wavelet transform to characterize the changes that occurred due to splicing (see Figure 5).

**3.3. DRLBP Histograms.** After extracting the chroma channels from sample image, the next step is to extract the discriminate patterns and estimate their distribution. For this purpose, we adopt DRLBP descriptor, which extracts the histogram of local binary patterns such as edges, corners, spots, and lines in the form of LBP codes. Then we approximate the distribution of these patterns while considering the local gradient magnitude at subsequent locations. DRLBP descriptor highlights the changes in local regions while considering the amount of change. The overview of DRLBP descriptor is given in equations (1) to (4), whereas the detailed description of DRLBP is presented in [19]. The binary patterns of each pixel from a  $3 \times 3$  window with 8 neighbors are computed from sample image; then the weighted histogram of binary patterns is computed from each region as defined in equation (3).

$$W_{LBP}(i) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} G_{x,y} \delta(LBP_{x,y}, i), \quad (1)$$

$$\delta(j, i) = \begin{cases} 1, & j = i, \\ 0, & \text{otherwise.} \end{cases}$$

Here  $n = 2^8$  the number of bins to represent the sample image in the form of histograms of 256 distinct patterns.  $G_{x,y}$  is the gradient magnitude of central pixel  $(x, y)$  which demonstrates the contribution of the corresponding binary pattern with respect to the intensity of pixel wise local change.  $M \times N$  represents resolution of each specific frequency band. To eliminate the reverse effect both in background and in foreground, we computed the weighted histogram  $W_{RLBP}$  as follows:

$$W_{RLBP}(i) = W_{LBP}(i) + (2^8 - 1 - i, 0 \leq i \leq 2^7). \quad (2)$$

After calculating the RLBP, the histogram of weighted discriminative LBP was computed to enhance the discriminative effect of binary patterns as follows:

$$W_{DLBP}(i) = |W_{LBP} - (i)W_{LBP}(2^8 - 1 - i)|, \quad 0 \leq i \leq 2^7. \quad (3)$$

Finally, DRLBP descriptor is computed by concatenating the histogram of weighted RLBP and weighted DLBP of each local region as follows:

$$DRLBP = [W_{RLBP}, W_{DLBP}]. \quad (4)$$

**3.4. Computation of the DWT-DRLBP Descriptor.** After computing the local DRLBP patterns of each channel  $Ch \in \{Cb, Cr\}$  from all subbands  $sb \in \{LL, LH, HL, HH\}$ ,

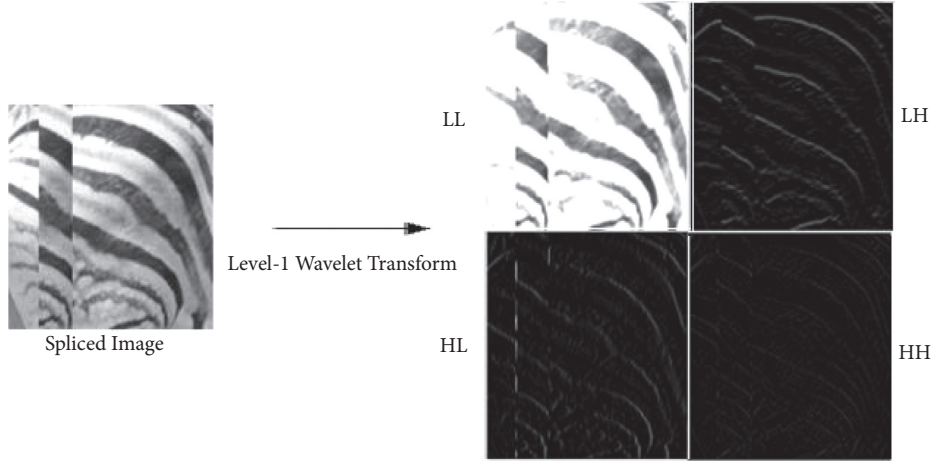


FIGURE 4: Decomposition of a given image into LL, LH, HL, and HH subbands using single level 2D DWT.

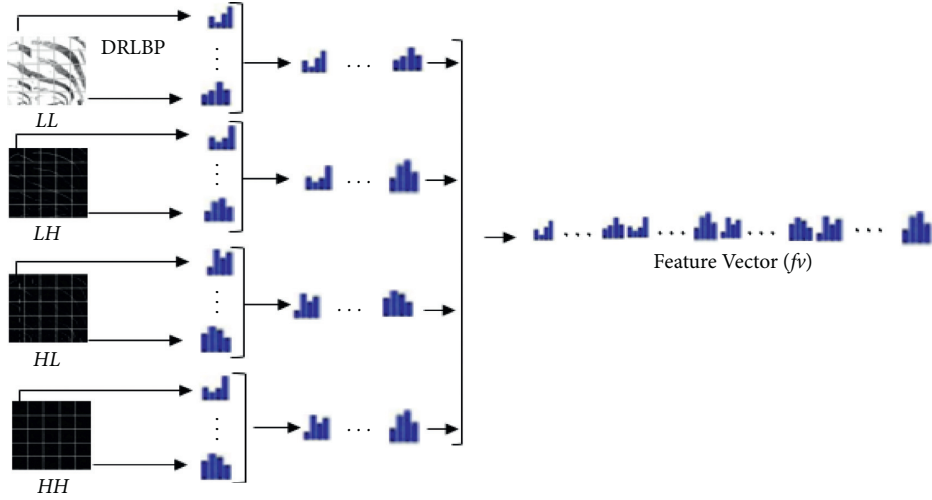


FIGURE 5: Encoding of spatially localized changes that occurred due to splicing using the DWT-DRLBP descriptor.

the histogram of all subbands is concatenated to form the DRLBP descriptor ( $fv$ ). The whole process of the computation of  $fv$  is given in Algorithm 1. The descriptor  $fv$  computes the overall structural changes without considering the spatial locations of image contents. The integration of the discriminative information and localized spatial changes into  $fv$  further enhances its discriminative potential. For this purpose, each channel of sample image is partitioned into  $K$  blocks (subblocks),  $B_1, B_2, \dots, B_K$ , each subblock with  $l \times m$  dimension such that  $K(l \times m) = M \times N$ . The descriptor  $fvB_i$  of each subblock  $B_i$  is computed.

At the end, the  $fvB_i$  of all subblocks is concatenated to represent the  $fv^{ch}$  of a channel with respect to each subband  $Sb \in \{LL, LH, HL, HH\}$  as represented in equation (6). Finally, the DWT-DRLBP descriptor is obtained to represent the sample image as described in

$$fv^{ch} = [fv^{LL}, fv^{LH}, fv^{HL}, fv^{HH}], \quad (5)$$

$$fv = [fv^{Cb}, fv^{Cr}]. \quad (6)$$

## 4. Performance Measures and Evaluation Methodology

To evaluate the performance of proposed image forgery detection technique three image databases were used. The performance evaluation techniques and datasets are described in this section.

**4.1. Dataset Description.** The performance of image forgery technique was evaluated using three datasets Columbia Color DVMM (DVMM) [11] and CASIA v1.0 and CASIA v2.0 benchmark datasets available publicly. Initially, we performed experiments on DVMM to evaluate the proposed method. CASIA v1.0 and CASIA v2.0 datasets were then used for further experiments and evaluation. DVMM dataset contains 183 authentic and 180 tampered images in TIFF format. The CASIA v1.0 dataset comprises 800 authentic and 921 spliced images. All tampered images are postprocessed using different geometric transformations. The CASIA v2.0 dataset contains 7,491 authentic and 5,123 forged images. We also evaluated the performance of proposed technique

on the combined dataset (the collection of abovementioned three datasets) to represent the generalization of proposed image forgery detection technique.

**4.2. Evaluation Policy.** The parameters of SVM were tuned with respect to the training dataset. We achieved the best performance with RBF kernel. The RBF filter depends on further two parameters which are *regularized coefficient* and *gamma*. The performance of RBF filter entirely depends on the optimal combination of these two parameters. The *regularized coefficient* performs an important role in balancing the complexity of the model by achieving highest forgery detection accuracy, whereas the *gamma* parameter in RBF kernel is used to define the nonlinear mapping between two points; in case of lower *gamma* value the far away points are considered as closest points. For image splicing detection we tuned the RBF kernel with  $2^5$  and  $2^{-5}$  for regularized coefficient and gamma, respectively, using grid-search method [31, 32]. We employed 10-fold cross validation in which the forged and authentic images are randomly divided into 10 folds of equal size. Ten performance measure values corresponding to the 10-folds are calculated and their average along with standard deviation (*std*) is reported as the performance of the system. The same procedure is repeated for each dataset.

**4.3. Performance Measures.** For evaluation, the forged images are considered as positive class while the authentic images are considered as negative class. We adopted the following performance evaluation techniques: accuracy, sensitivity, specificity, and false positive rate. Accuracy is the percentage of samples accurately predicted as forged or authentic to the total number test images, computed as follows:

$$\text{ACC} = \frac{(\text{TP} + \text{TN})}{\text{TP} + \text{TN} + \text{FN} + \text{FP}} \times 100\%. \quad (7)$$

Here the symbol *TP* characterizes the number of samples that are forged, and the classifier also predicted them as forged. The symbol *TN* represents the number of images that are authentic, and the classifier also predicted them as authentic. Moreover, the symbol *FP* represents the number of images that were authentic and classifier predicted them as forged, and *FN* represents the number of images that were forged and the classifier predicted them as authentic.

**True Positive Rate.** The true positive rate is also called sensitivity, which represents the percentage of predicting a forged image as forged, calculated as

$$\begin{aligned} \text{TPR} &= \text{SN} \\ &= \frac{\text{TP}}{\text{TP} + \text{FN}} \times 100. \end{aligned} \quad (8)$$

**True Negative Rate.** The true negative rate is also called specificity, which represents the percentage of predicting a genuine image as genuine, computed as

$$\begin{aligned} \text{TNR} &= \text{SP} \\ &= \frac{\text{TN}}{\text{TN} + \text{FP}} \times 100. \end{aligned} \quad (9)$$

**False Positive Rate.** The false positive rate represents the percentage of predicting the sample images as forged which are actually misclassified as authentic.

$$\text{FPR} = (1 - \text{TNR}) \times 100\%. \quad (10)$$

**Parameter Tuning.** The proposed system involves many parameters. Figure 5 illustrates the participating parameters. Tuning the parameters in a thorough manner to find the optimal set is not an easy task, which is considered as an optimization problem. From a practical point of view parameter setting is important. We determined empirically various parameter settings in this paper. After parameters tuning, the best parameters values used by proposed method are shown in Table 1.

## 5. Experimental Results and Discussion

The performance of proposed method on different benchmark datasets is given in Table 2. The ROC curves are shown in Figure 6. To evaluate the performance of image splicing techniques developed in this research, we applied the proposed technique on DVMM dataset. The DVMM dataset contains 180 normal images and 183 forged color images. The forged images are produced by tampering the authentic images by applying the crop-paste method of the vertical and horizontal strips.

**5.1. Robustness on Geometric Transformations.** Geometric transformations such as scaling (resizing), rotation, and deforming are applied normally in combination or individually on spliced regions to hide the cues of forgery. These transformations are applied on spliced region(s) in CASIA v1.0 and CASIA v2.0 datasets. Figure 7 shows the accuracy of the method against these transformations. When geometric transformations are applied on spliced region(s), the changes along the boundary turn into sharp edges (splicing artifacts), which are needed to be modeled properly. In general, the method performs well with respect to different geometric transformation, because splicing artifacts are modeled properly by DWT-DRLBP descriptor.

**5.2. Robustness on Spliced Region(s) Size.** Splicing regions are detected by exploring variations of intrinsic features, which are usually consistent in unaltered images. The method is explored on small, medium, and large spliced region(s). Figure 8 shows the results with these sizes. It is fact that local inconsistencies of spliced region(s) are useful in exposing forgery, which is exposed effectively using DTW-DRLBP descriptor.

- (1) RGB image  $I$ , the number  $K$  of blocks
  - a) Convert  $I$  to YCbCr
  - b) DWT-DRLBP descriptor  $f_v$
- (2) for each channel  $Ch \in \{Cb, Cr\}$  of image  $I$
- (3) a) Apply 2D-DWT on  $Ch$ ,  $Sb \in [LL, LH, HL, HH]$
- (4) end for
- (5)  $S \mathbf{b} = \mathbf{H}\mathbf{H}$
- (6) Divide  $S \mathbf{b}$  into  $K$  blocks:  $\mathbf{B}_1, \mathbf{B}_2, \dots, \mathbf{B}_K$
- (7) for each block  $\mathbf{B}_k, k = 1, 2, \dots, k$
- (8) Compute DRLBP histogram  $f_v^{sb}$
- (9)  $f_v^{Sb} = [f_v^{Sb_1}, f_v^{Sb_2}, \dots, f_v^{Sb_K}]$
- (10)  $f_v^{Ch} = [f_v^{LL}, f_v^{LH}, f_v^{HL}, f_v^{HH}]$
- (11)  $f_v = [f_v^{Cb}, f_v^{Cr}]$
- (12) end for

ALGORITHM 1: Computation of DWT-DRLBP.

TABLE 1: The optimal parameters set of the proposed method.

Preprocessing	Color channel/s Block division	Cb and Cr Nonoverlapped
	P	8
DRLBP	R	1
	Mapping type	$u2$
Classification	SVM kernel	RBF

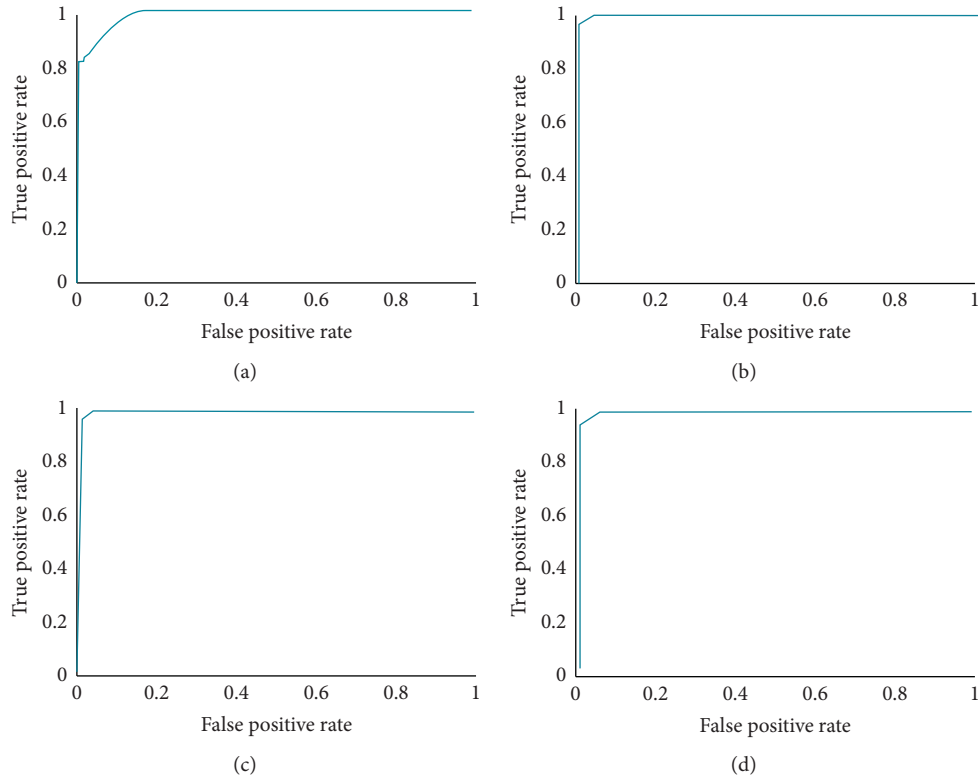


FIGURE 6: ROC curves on datasets: (a) DVMM, (b) CASIA v1.0, (c) CASIA v2.0, and (d) combined.

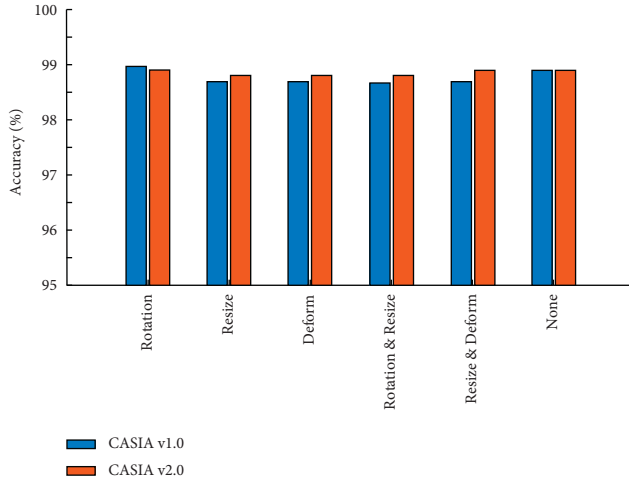


FIGURE 7: The proposed method accuracy on geometric transformations.

**5.3. Robustness on Spliced Region(s) Shape.** Splicing of different types of shapes is very common to gain illegal benefits. This section explores the effect of spliced region shapes on the performance. CASIA v1.0 and CASIA v2.0 datasets contain the four region shapes of the spliced part(s) that are circular (CR), rectangular (RECT), triangular (TRI), and arbitrary (ARB). Figure 8 shows the results with these shapes. The method is robust on different shapes of spliced region(s).

**5.4. Robustness on JPEG Images.** The performance of proposed technique was evaluated on the JPEG images taken from CASIA v1.0 and CASIA v2.0 datasets and achieved 98% accuracy. JPEG compression is performed by applying DCT quantization. When splicing is performed in JPEG images the forgery occurs in the form of block mismatching and blocks are not aligned with their neighbors which causes extraneous edges. The proposed techniques achieved high forgery detection accuracy because it properly explored the inconsistencies in the spliced image using DWT-DRLBP descriptor.

**5.5. Comparison with State-of-the-Art Methods.** In this section we compared the results achieved in this research with the state-of-the-art image splicing technique for image forgery detection. Table 3 demonstrates the performance of state-of-the-art methods only on the corresponding datasets, because they achieved best results either on CASIA v1.0 or CASIA v2.0 or DVMM. It demonstrates that the proposed technique outperforms the state-of-the-art techniques with respect to every performance evaluation criterion. It can also be noted that the proposed approach performed significantly on combined dataset, which witnessed the generalization of the proposed technique. Overall, the method has high accuracy and true positive rate, while maintaining low false positive rate as compared to other state-of-the-art methods.

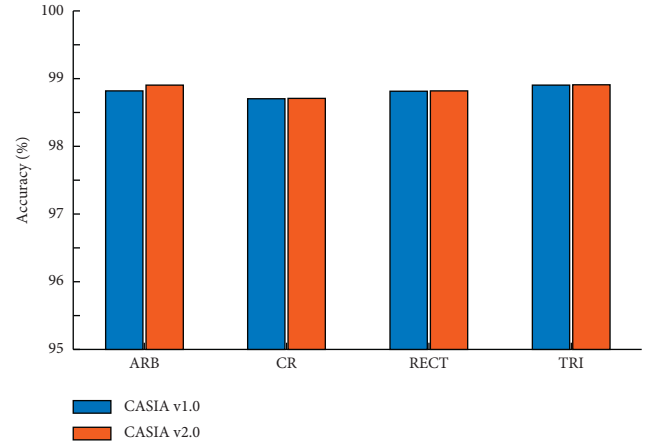


FIGURE 8: The proposed method accuracy on spliced region(s) shape.

TABLE 2: Performance of image forgery detection.

Dataset used	ACC	TPR	FPR	AUC
DVMM	97.21	97.87	3.88	0.98
CASIA v1.0	98.59	99.34	1.24	0.99
CASIA v2.0	98.88	99.32	0.96	0.99
Combined	98.95	99.91	2.05	0.99

TABLE 3: Comparison with recent state-of-the-art image splicing detection methods.

Approaches	Dataset used	ACC	TPR	FPR	AUC
Proposed	DVMM	97.21	97.87	3.88	0.98
	CASIA v1.0	98.59	99.34	1.24	0.99
	CASIA v2.0	98.33	97.32	0.96	0.99
	Combined	98.95	99.91	2.05	0.99
[5]	DVMM	96.39	97.92	4.46	0.97
	CASIA v1.0	94.89	92.30	2.77	0.94
[31]	CASIA v2.0	97.33	98.50	3.47	0.97
	DVMM	91.14	93.07	16.14	—
[32]	CASIA v1.0	96.17	97.65	6.84	—
	CASIA v2.0	97.86	98.82	2.79	—
[6]	Composite	93.21	—	—	—
	CASIA v1.0	90.18	93.00	2.11	—
[33]	CASIA v2.0	96.21	93.00	2.90	—
	Composite	94.64	93.00	7.20	—
[34]	CISE	93.36	92.99	1.89	—
	CASIA v1.0	94.29	—	—	0.93
	CASIA v2.0	96.52	—	—	0.97
[18]	DVMM	94.17	—	—	0.93
	CASIA v1.0	95.4	—	—	—
	CASIA v2.0	95.6	—	—	—
[19]	DVMM	94.8	—	—	—
	CASIA v1.0	95.4	—	—	—
	CASIA v2.0	95.6	—	—	—
[35]	DVMM	94.8	—	—	—
	CASIA v2.0	95.4	—	—	—
[20]	CASIA v2.0	99.54	95	—	—

## 6. Conclusion

To represent a test image for authentication, we employed DWT-DRLBP descriptor for feature extraction. Chroma components of a test image are decomposed into subbands

using single level DWT to exploit splicing inconsistencies. A robust texture descriptor DRLBP is employed to capture the detailed statistics from these subbands. DWT-DRLBP descriptor makes the proposed method capable enough to detect splicing image forgery even in the presence of postprocessing operations. Performance of DWT-DRLBP descriptor is obtained and examined by employing SVM classifier with 10-fold cross validation. Three publicly available benchmark datasets were used for experiments and evaluation. Our proposed method achieved 98.95% accuracy on combined dataset and is robust as compared to other state-of-the-art methods. These results also endorsed the success and robustness of DWT-DRLBP descriptor, used to model inconsistencies in images caused by splicing forgery. Furthermore, SVM with RBF kernel classified any given image as authentic or spliced and finally ensured the excellent accuracy of results. The future work is to localize the spliced region(s) to boost the trust on results and to measure the degree of splicing.

### Data Availability

Data used for this study and simulation will be provided upon request to the reviewer for validation.

### Conflicts of Interest

The authors declare that they have no conflicts of interest regarding the present study.

### Acknowledgments

The authors extend their appreciation to the Deputyship for Research & Innovation, Ministry of Education in Saudi Arabia, for funding this work through Project no. "375213500." Also, the authors would like to extend their sincere appreciation to the central laboratory at Jouf University for supporting this study.

### References

- [1] M. K. Abdolmaleki, M. S. Riasi, M. Enayati et al., "A digital imaging method for evaluating the kinetics of vapo-chromic response," *Talanta*, vol. 209, Article ID 120520, 2020.
- [2] N. L. Iacobici, M. I. Frigura, H. E. Filipescu, M. Nen, F. M. I. Frigura, and M. Iorga, "Digital imaging processing and reconstruction for general applications," in *Proceedings Of the IEEE 18th World Symposium on Applied Machine Intelligence and Informatics (SAMII)*, pp. 231–234, Herl'any, Slovakia, January 2020.
- [3] A. Khurshid, H. Zulfiqar, and H. Muhammad, "Copy-move and splicing image forgery detection and localization techniques: a review," *Australian Journal of Forensic Sciences*, vol. 49, no. 3, pp. 281–307, 2017.
- [4] X. Zhao, J. Li, S. Li, and S. Wang, "Detecting digital image splicing in chroma spaces," in *Proceedings of the International workshop on digital watermarking*, pp. 12–22, Berlin, Germany, October 2010.
- [5] P. Xunyu, "Digital image forensics with statistical analysis," *Handbook Of Digital Forensics of Multimedia Data and Devices*, John Wiley & Sons, NJ, USA, 2015.
- [6] E. Mahmoud and Z. Hongli, "Two-stage keypoint detection scheme for region duplication forgery detection in digital images," *Journal of Forensic Sciences*, vol. 63, no. 1, pp. 102–111, 2018.
- [7] G. Muhammad, M. H. A. Hammadi, M. Hussain, and G. Bebis, "Image forgery detection using steerable pyramid transform and local binary pattern," *Machine Vision and Applications*, vol. 25, no. 4, pp. 985–995, 2014.
- [8] J. Goh and V. L. L. Thing, "A hybrid evolutionary algorithm for feature and ensemble selection in image tampering detection," *International Journal of Electronic Security and Digital Forensics*, vol. 7, no. 1, pp. 76–104, 2015.
- [9] A. A. Alahmadi, M. Hussain, H. Aboalsamh, G. Muhammad, and G. Bebis, "Splicing image forgery detection based on DCT and Local Binary Pattern," in *Proceedings of the IEEE Global Conference on Signal and Information Processing*, pp. 253–256, Austin, TX, USA, September 2013.
- [10] G. H. Min and H. H. Dong, "Identification method for digital image forgery and filtering region through interpolation," *Journal of Forensic Sciences*, vol. 59, no. 5, pp. 1372–1385, 2014.
- [11] W. Jinwei, N. Qiye, L. Guangjie, L. Xiangyang, and K. J. Sunil, "Image splicing detection based on convolutional neural network with weight combination strategy," *Journal Information Security and Applications*, vol. 54, pp. 1–8, 2020.
- [12] A. Costanzo, I. Amerini, R. Caldelli, and M. Barni, "Forensic analysis of SIFT keypoint removal and injection," *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 9, pp. 1450–1464, 2014.
- [13] T. T. Ng, S. F. Chang, and Q. Sun, "A Data Set of Authentic and Spliced Image Blocks," *ADVENT Technical Report #203-2004-3*, pp. 6–9, Columbia University, NY, USA, 2004.
- [14] J. Dong and W. Wang, "CASIA image tampering detection evaluation databases," in *Proceedings of the Signal and Information Processing*, pp. 422–426, Beijing, China, 2015.
- [15] T. T. Ng, S. F. Chang, and Q. Sun, "Blind detection of photomontage using higher order statistics," in *Proceedings of the international symposium on circuits and systems*, pp. 688–691.
- [16] F. Hsu and S. Chang, "Detecting image splicing using geometry invariants and camera characteristics consistency," in *Proceedings of the IEEE International Conference on Multimedia and Expo*, pp. 549–552, Toronto, ON, Canada, 2006.
- [17] Y. Q. Shi, C. Chen, and W. Chen, "A natural image model approach to splicing detection," in *Proceedings of the 9th ACM Workshop on Multimedia & Security*, pp. 51–62.
- [18] W. Wang, J. Dong, and T. Tan, "Image tampering detection based on stationary distribution of Markov chain," in *Proceedings of the 17th IEEE international conference on image processing*, pp. 2101–2104, Hong Kong, China, December 2010.
- [19] X. Zhao, S. Li, S. Wang, J. Li, and K. Yang, "Optimal chroma-like channel design for passive color image splicing detection," *EURASIP Journal on Applied Signal Processing*, vol. 2012, pp. 1–11, 2012.
- [20] A. Alahmadi, M. Hussain, H. Aboalsamh, G. Muhammad, G. Bebis, and H. Mathkour, "Passive detection of image forgery using DCT and local binary pattern," *Signal, Image and Video Processing*, vol. 11, no. 1, pp. 81–88, 2017.
- [21] N. T. Pham, J. Lee, G. Kwon, and C. Park, "Efficient image splicing detection algorithm based on markov features," *Multimedia Tools and Applications*, vol. 78, no. 9, Article ID 12405, 2019.



- [22] H. Jalab, T. Subramaniam, R. Ibrahim, H. Kahtan, and N. Noor, "New texture descriptor based on modified fractional entropy for digital image splicing forgery detection," *Entropy*, vol. 21, no. 4, pp. 371–385, 2019.
- [23] R. C. Gonzalez and R. E. Woods, *Digital image processing* Vol. 4, Addison-Wesley, MA, USA, 1992.
- [24] R. Salloum, Y. Ren, and C.-C. Jay Kuo, "Image splicing localization using a multi-task fully convolutional network (mfcn)," *Journal of Visual Communication and Image Representation*, vol. 51, pp. 201–209, 2018.
- [25] X. Bi, Y. Wei, B. Xiao, and W. Li, "Rru-net: the ringed residual u-net for image splicing forgery detection," in *Proceedings of the the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, June 2019.
- [26] X. Wang, H. Wang, S. Niu, and J. Zhang, "Detection and Localization of Image Forgeries Using Improved Mask Regional Convolutional Neural Network," *Mathematical Biosciences and Engineering*, vol. 16, pp. 4581–4593, 2019.
- [27] Y. Wu, W. Abd-Almageed, and P. Natarajan, "Busternet: detecting copy-move image forgery with source/target localization," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 168–184, Munich, Germany, September 2018.
- [28] A. Satpathy, X. Xudong Jiang, and H. L. How-Lung Eng, "LBP-based edge-texture features for object recognition," *IEEE Transactions on Image Processing*, vol. 23, no. 5, pp. 1953–1964, 2014.
- [29] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.
- [30] C. C. Chang and C. J. Lin, "LIBSVM: a library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 27, no. 3, pp. 21–27, 2011.
- [31] C. W. Hsu, C. C. Chang, and C. J. Lin, *A Practical Guide to Support Vector Classification*, National Taiwan University, Taipei, Taiwan, 2003.
- [32] M. Sokolova, N. Japkowicz, and S. Szpakowicz, "Beyond accuracy, F-score and ROC: a family of discriminant measures for performance evaluation," in *Proceedings of the Australasian joint conference on artificial intelligence*, pp. 1015–1021, Berlin, Germany, January 2006.
- [33] X. Zhao, S. Wang, S. Li, and J. Li, "Passive image-splicing detection by a 2-D noncausal Markov model," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 25, no. 2, pp. 185–199, 2014.
- [34] M. Hussain, S. Qasem, G. Bebis, G. Muhammad, H. Aboalsamh, and H. Mathkour, "Evaluation of image forgery detection using multi-scale weber local descriptors," *The International Journal on Artificial Intelligence Tools*, vol. 24, no. 4, pp. 1–28, 2015.
- [35] T. H. Park, J. G. Han, Y. H. Moon, and I. K. Eom, "Image splicing detection based on inter-scale 2D joint characteristic function moments in wavelet domain," *EURASIP Journal on Image and Video Processing*, vol. 2016, no. 1, pp. 30–39, 2016.

## Research Article

# StFuzzer: Contribution-Aware Coverage-Guided Fuzzing for Smart Devices

Jiageng Yang, Xinguo Zhang, Hui Lu , Muhammad Shafiq, and Zhihong Tian 

*Cyberspace Institute of Advanced Technology, Guangzhou University, Guangzhou, China*

Correspondence should be addressed to Zhihong Tian; [tianzhihong@gzhu.edu.cn](mailto:tianzhihong@gzhu.edu.cn)

Received 20 July 2021; Revised 21 August 2021; Accepted 30 August 2021; Published 20 September 2021

Academic Editor: Muhammad Ahmad

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

The root cause of the insecurity for smart devices is the potential vulnerabilities in smart devices. There are many approaches to find the potential bugs in smart devices. Fuzzing is the most effective vulnerability finding technique, especially the coverage-guided fuzzing. The coverage-guided fuzzing identifies the high-quality seeds according to the corresponding code coverage triggered by these seeds. Existing coverage-guided fuzzers consider that the higher the code coverage of seeds, the greater the probability of triggering potential bugs. However, in real-world applications running on smart devices or the operation system of the smart device, the logic of these programs is very complex. Basic blocks of these programs play a different role in the process of application exploration. This observation is ignored by existing seed selection strategies, which reduces the efficiency of bug discovery on smart devices. In this paper, we propose a contribution-aware coverage-guided fuzzing, which estimates the contributions of basic blocks for the process of smart device exploration. According to the control flow of the target on any smart device and the runtime information during the fuzzing process, we propose the static contribution of a basic block and the dynamic contribution built on the execution frequency of each block. The contribution-aware optimization approach does not require any prior knowledge of the target device, which ensures our optimization adapting gray-box fuzzing and white-box fuzzing. We designed and implemented a contribution-aware coverage-guided fuzzer for smart devices, called StFuzzer. We evaluated StFuzzer on four real-world applications that are often applied on smart devices to demonstrate the efficiency of our contribution-aware optimization. The result of our trials shows that the contribution-aware approach significantly improves the capability of bug discovery and obtains better execution speed than state-of-the-art fuzzers.

## 1. Introduction

Various smart devices have been installed in many situations. The security of smart devices is essential to ensure the normal functions of these smart devices. The root cause of any attack is the potential vulnerabilities in smart devices. Finding potential bugs in smart devices is the most important factor to ensure the security of smart devices. There are many automatic approaches to identify bugs in applications or devices, such as taint, symbolic execution, or fuzzing. Fuzzing is the most effective vulnerability identification technology, which inputs various random data to the target application to anticipate a dangerous execution state. If the test case crashes the target application, a bug must be triggered. However, fuzzing mainly recognizes bugs in the shallow logic of an application, because modern applications

usually require complex input formats that are not conducive to exploring deeper execution states. Fuzzing randomly mutates an input file that would destroy the crucial data structure of the input file. The mutated file as a test case will cause the target application to stop in a shallow execution state.

State-of-the-art coverage-guided fuzzing such as AFL [1] utilizes edge coverage to select test cases which triggered more interesting execution states. If a test case triggered a new branch, it will be regarded as an interesting seed. The natural branch coverage mechanism thinks any branch of a target application has the same contribution. However, branches that belong to deep logic are much more difficult to trigger than branches that exist in shallow logic [2]. Especially, in the smart device, the execution of these devices involves multiple sensors and arguments, that the execution

must be satisfied by special sensors or argument. In other words, triggering branches in deep logic contributes more than branch being triggered in shallow logic. However, AFL-family cannot consider the contribution of different branches. In detail, fuzzing that employs branch coverage preserves all test cases which triggered new branches as seeds. These seeds will be used to generate new test cases in next iteration. Unfortunately, AFL cannot distinguish branches which have more interesting states. Therefore, AFL has no effective strategy for seed selection. In order to optimize seed selection strategy of coverage-guided fuzzing, there are two problems should be solved. First, we should distinguish the contributions of different branches according to the depth of a branch. Second, we should select test cases with greater contributions for the next fuzzing iteration.

There are many works related with seed selection strategy. Rebert et al. [3] proposed that good seed selection strategies can improve the speed of fuzzing and increasing the chance of recognizing vulnerabilities, even reducing the runtime and memory overhead of fuzzing. AFLFast [4] found low-frequency paths through Markov chain and prioritized seed triggered these paths. AFLFast exposed several previously unreported CVEs that could not be exposed by AFL in 24 hours. The seed selection strategies of AFLFast increase the exploration frequencies of low-frequency paths successfully. VUzzer believed that deeper code blocks are more difficult to be triggered and prioritized seeds exercising longer path. VUzzer found more crashes than AFLPIN [5], which indicated that exploring deeper paths may increase the probability of finding crash. Good seed selection strategies can improve the ability to find bugs, but it is often difficult to determine the precise contributions of various test cases.

In this paper, we present StFuzzer, a contribution-aware coverage-guided fuzzing which obtains precise contributions of all branches and guides fuzzing to explore deeper execution states. Comparing with other seed selection solutions such as VUzzer, our work provides a solution based on the contribution information that combines static and dynamic information, which considers the accurate control flow information and dynamic runtime information. This combination scheme can help tremendously to improve the efficiency of path exploration and avoid unnecessary drilling. The crucial intuition of our work is that deeper basic blocks can contribute more to exploring interesting execution states. The most important question is how to calculate the depth of each basic block. We defined the depth of basic blocks based on the control flow feature of target applications and modify the depth information according to the runtime feature of fuzzing process. The control flow feature of the target application will be captured by the customized static analysis. Our solution dynamically collects useful runtime information to evaluate the value of different paths. These static analysis and dynamic information guide the fuzzing process to deeper code region and to trigger more interesting execution states. Due to this novel combination solution in seed selection, StFuzzer could find more crashes than AFL on various real-world applications. Meanwhile,

our solution reduces the useless exploration of shallow logic, and StFuzzer reproduced the same crashes faster than AFL.

*Contribution.* This paper makes the following contributions:

- (1) We propose a novel contribution-aware seed selection solution that combines the accurate control flow information of static analysis and dynamic runtime information to evaluate the contributions of each branch.
- (2) We design a sophisticated contribution calculation algorithm that statically assesses the value of various branches. And this algorithm considers the influence of the loop structure, which is unresolved in similar solutions.
- (3) We implement our approach to build a fuzzer tool named StFuzzer based on AFL and evaluate StFuzzer on 4 real-world applications running on smart devices.
- (4) StFuzzer found 5 zero-day vulnerabilities and already reported them to corresponding vendors.

## 2. Background and Related Work

In this section, we present the background of coverage-guided fuzzing. Firstly, we introduce the main workflow of coverage-guided fuzzing. Secondly, we describe the details and limitations of seed selection and seed mutation in fuzzing loop.

*2.1. Coverage-Guided Fuzzing.* Fuzzing is an automatic software testing technique that attempts to input random data into the target application and expects the target has exceptions. If the fuzzing process captures an exception, it means that a vulnerability had been triggered by a test case. The fuzzing technique can be classified as generation-based and mutation-based. The generation-based fuzzing relies on grammar constraints to generate random inputs, such as Peach [6]. Since generation-based fuzzers must require grammar constraints of corresponding input format, the usage of generated fuzzing is not universal. Mutation-based fuzzers mutate seeds of the initial corpus to generate test cases which are put into the target application for execution. Mutation-based fuzzers do not require configuration files for different target application, which makes this technology more versatile.

According to different exploration strategies, mutation-based fuzzing can be classified into directed fuzzing and the coverage-guided fuzzing. Directed fuzzing will explore more specific code regions based on the strategy used. Directed gray-box fuzzing [7] requires researchers provide deterministic strategies for specified types of vulnerabilities. Therefore, directed fuzzing is available for finding defined bugs, but it is unfair for unknown bugs. Coverage-guided fuzzing is the most popular technology for finding vulnerabilities. Coverage-guided fuzzing determines the contribution of test cases based on code coverage. The intuition of coverage-guided fuzzing is that the greater the coverage, the greater the chance of triggering bugs. Figure 1 shows the

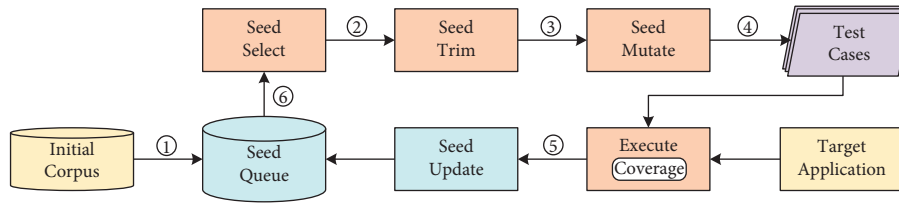


FIGURE 1: The main workflow of coverage-guided fuzzing.

main workflow of coverage-guided fuzzing. It usually contains six stages as follows: (1) it reads the initial corpus to load original test cases into the seed queue, (2) scheduling higher-priority seeds from seed queue as input files, (3) trimming input files to the smallest size and coverage of the trimmed file is the same as the file before trimming, (4) mutating the input file through various heuristic mutation algorithms to generate the test case, (5) monitoring coverage triggered by test cases and persisting test cases with new coverage, and (6) going to Step 2 and starting the next loop iteration.

Steps 2 to 6 build a complete fuzzing loop iteration. The core components include seed selection, seed mutation, and execution monitor. Any improvements of these components would promote the ability of coverage-guided fuzzing to find vulnerabilities. In following sections, we will analyze and discuss the core components of the fuzzing loop in detail.

**2.2. Seed Selection.** Seed selection module schedules more interesting seeds according to some runtime features determined by the employed policies. Some studies demonstrated that good seed selection algorithms can greatly increase the number of bugs found. Rebert et al. proposed that good seed selection strategies can improve the speed of fuzzing and reduce the runtime and memory overhead of fuzzing process. AFLFast found low-frequency paths through Markov chain and prioritized seed triggered these paths. VUzzer believed that deeper basic blocks are more difficult to be triggered and prioritized seeds which traversed deeper blocks. Meanwhile, VUzzer observed that the most of generated test cases will end with error handling code and deprioritized seeds trapped into error handling code. CollAFL [8] proposed two selection policies which drive the fuzzer towards nonexplored paths. CollAFL prioritized seeds that invoke more memory access operations or paths have more nonexplored neighbor branches. Wang et al. [9] proposed a seed scheduling algorithm to support the multilevel coverage metric based on the multiarmed bandit model, which archives higher code coverage.

Other seed selection solutions attempt to guide the fuzzing to explore specific vulnerable code regions [10, 11]. QTEP [12] presented a quality-aware seed prioritization technique, which leveraged the statistic defect prediction model to defect fault. ProFuzzer [13] proposed an on-the-fly probing technique that discovered the relations between input bytes and program behaviors, which utilizes the information to select more powerful seeds. GREYONE [14] utilized lightweight dynamic taint analysis to evaluate the constraint conformance on all tainted untouched branches,

and prioritizing seeds which could reach these branches. FIFUZZ [15] proposed a context-sensitive SFI-based approach to guide fuzzing exploring error handling code.

There are ample works on seed selection, and the result of these studies indicates that seed selection is crucial for coverage-guided fuzzing. However, when we reproduced some trials through these open source works, we found two problems. On the one hand, the implementation of some studies is inadequate. The implementation proposed by the work cannot exactly perform the corresponding selection policy. For example, as aforementioned, VUzzer proposed prior exploring deeper block policy. The solution of VUzzer measured the depth of basic blocks through the Markov probabilistic model, but it simply assigns a fixed probability of 1 to each backedge belonging to the loop structure. In general, the loop structure has a great influence on the depth of a basic block. In a word, its seed selection algorithm does not consider the loop structure in the application, resulting in inaccurate results of the fuzzing process. On the other hand, implementing some policies requires not only static compile-time information but also dynamic runtime information. For example, VUzzer assigned an equal probability to all basic blocks' outgoing edges. Remarkably, it is not possible that outgoing edges of a basic block have the same probability to be executed. The probability of a branch should be calculated by statistics. Therefore, the imprecise probability makes VUzzer's result incorrect. In conclusion, the accuracy of path features has a negative impact due to the imperfect implementation.

Following the observations, we propose a novel contribution mechanism, which incorporated the influence of the control flow and the dynamic statistics at the runtime. Our solution considers the accurate control flow information which solves the challenge caused by nested branches in various loop structures. And we also convert the triggered frequencies of basic blocks during the iterations of the fuzzing process into the contribution of each block. Compared with other solutions, the contribution mechanism measures the distances of all basic blocks for the program exploration, which is more accurate to seeds with high quality. And the contribution mechanism does not require prior knowledge of the target application that is appropriate for all applications.

**2.3. Seed Mutation.** Seed mutation module mutates seeds by multiple mutating algorithms and generates new test cases for the next loop iteration. Different mutation algorithms insert or replace special characters in the seed for different situations and attempt to traverse all the traps within inputs.

For example, AFL mutates seeds through four random mutation algorithms: bitflip, arithmetic, interest, and havoc.

However, there are some challenges in the target application, which restrain the fuzzing process to explore more code region. For example, most applications have a variable number of sanity checks. These sanity checks involve intricate inspections or require unique input formats, which are difficult to satisfy through arbitrary mutations. Good seed mutation strategies need to answer two questions: where to mutate and how to mutate.

Many solutions to both of these questions have been proposed. Program transformation-based approaches disable sanity checks to help the fuzzing process to discover more execution logic. MutaGen [16] proposed that mutating the code of generating programs and leveraged available information about the input format encode in the generated programs to produce high-coverage test cases. T-Fuzz [17] attempted to strip relevant code that hinders program explosion. However, changing the normal logic will introduce more unintended consequences, leading to more false positive bugs. And transformation-based approaches have to suffer trimming explosion in complex applications, which the blind mutation engine is limited for program explosion.

Hybrid fuzzing utilizes symbolic execution to solve complex branch which cannot be satisfied through simple mutation engine. However, symbolic execution or concolic execution produces huge amount of runtime overhead and memory overhead, which have hindered the efficiency of fuzzing. Ma et al. [18] studied an approach to automatically find program executions which reach a particular line. Driller [19] used concolic execution to produce test cases when the fuzzer getting stuck, and got higher coverage. However, driller has a terrible execution speed and works with a low efficiency. The best way to mitigate the performance overhead is proposing a lightweight symbolic execution engine. QSYM [20] designed a fast concolic execution engine and implemented instruction-level symbolic emulation to overcome performance bottlenecks of the concolic executor. Some works believed that reducing the calls of symbolic execution will significantly mitigate overhead. Digfuzz [21] proposed a probabilistic path prioritization model to estimate the probability of exercising each path and prioritized them for concolic execution. Peng et al. [22] leveraged a distance-based directed fuzzing and a dominator-based directed symbolic execution mechanism to discover 1-day vulnerabilities in binary patches. SAVIOR [23] prioritized the concolic execution of the seeds which can uncover more vulnerabilities. Lee et al. [24] proposed a constraint-guided directed fuzzing based on the target site and the data conditions. However, it is difficult to evaluate whether a seed uncovers more vulnerabilities if lacking priori knowledge.

Some fuzzers depended on the result of static or dynamic analysis, such as honggfuzz [25] and VUzzer. Honggfuzz recognized operands of CMP-like instructions at runtime to insert into the seed. VUzzer leveraged control- and data-flow features of the application to consider interesting factors for mutation. Steelix [26] proposed a program-state based approach, which utilized lightweight static analysis to provide

comparison progress information to infer magic bytes efficiently. Learn and Fuzz [27] attempted to capture the structure of well-formed inputs and utilize well-structured seed-based deep learning. Rajpal et al. designed DNN (deep neural network) solution to predicate which bytes in the seed to mutate. FuzzGuard [28] utilized the learned model to predict a given input if shortened the seed distance to improve the effectiveness of directed fuzzing. Some works [28–34] designed semantic models to optimize mutation or identify potential vulnerabilities or attacks based on special target platforms. Liang et al. [35] proposed a secure decision tree classification for online diagnosis services on different platforms. However, according to evaluation of trials, the result of various lightweight static analysis has a slight effect on program explosion. Meanwhile, the effects of learning-based optimizations are also unclear, because of the uninterpretable results of learning.

As aforementioned, multiple optimizations for seed mutation involve symbolic or concolic execution, static analysis, dynamic taint, or machine learning. The hybrid fuzzing will be faced on the path explosion program and heavyweight performance overhead, which seriously reduce the effectiveness of the fuzzing process. The taint-assisted fuzzing also has numerous performance overhead and the overtaint or undertaint problems. The learning-based optimization is unclear or uninterpretable for fuzzing. In a word, existing optimizations cannot improve the effectiveness of seed mutation.

### 3. Calculation of Contribution

In a large-scale application, the control flow graph of the application may be very complicated, resulting in a low coverage of the fuzzing process. In the fuzzing process, most test cases traverse the simple logic of applications and the complex logic will be rarely triggered. Unfortunately, the more complex the logic, the more bugs. Because it is difficult to discover complex logic by the fuzzing process, simple logic is easier to be triggered, and a lot of time is spent exploring simple logic. The contribution of each basic block for exploring the whole application is different. Basic blocks in complex logic have more contribution than basic blocks in simple logic. The existing approach did not apply the contribution of basic blocks or thought that all basic blocks have the same contribution. In this paper, for each basic block, our solution gives quantified contributions based on control flow and execution flow. And we apply the contribution as an important factor for the fuzzing process.

*3.1. Static Contribution of Basic Block.* Due to different control flow structures, basic blocks of the target application have different execution probabilities. The entry block of a called function must be executed, so the execution probabilities of entry block are 1. We cannot accurately calculate the jumping probabilities of a branch that a basic block jumps to any successor blocks. Therefore, for a given basic block, we assign equal jumping probabilities to all its successor blocks. In other words, if a basic block B has  $n$

successors, the jumping probabilities of any successors are  $1/n$ . As same as the transition of VUzzer, the execution probability of each basic block will be dependent on a probabilistic model called Markov process. The execution probability of a basic block hinges on the execution probability of its predecessor block and its own jumping probability. The execution probability of a basic block B is calculated as follows:

$$\text{prob}(B) = \sum_{p \in \text{pred}(B)} \text{prob}(pB) * \text{prob}j(B), \quad (1)$$

where  $\text{pred}(B)$  is the set of all the predecessors of B and the  $\text{prob}j(B)$  is the jumping probability of B. In this way, we can regard the whole execution as a Markov chain to evaluate the contribution of each node in the CFG.

However, this solution is only applicable to the special case where the CFG is a directed acyclic graph and cannot deal with the explicit or implicit loop structures. To be specific, loop statements or jump statements would lead to cyclic structures in the CFG. The cyclic structure makes certain control flow within loop structures be repeatedly executed, which disrupts the calculation by the Markov chain. As aforementioned, the jumping probability of any block must be less than one. According to formula (1), the execution probability of the header block in a loop structure will become smaller and smaller with multiple iterations. The header block dominates all blocks of the loop structure, so the execution probability of the header block must become larger and larger with multiple iterations, which is contrary to the result of calculation through the Markov chain. Obviously, the execution probability inferred by the Markov chain is unsuitable to the real consequence.

Considering the above problems, we utilize mathematical expectation to estimate the static contribution feature of loop structures instead of execution probabilities. In loop structures, the execution probability of a basic block is an integral multiple of the execution probability within a separate iteration. Since the header block dominates other basic blocks in the loop, the execution probability of other blocks is similar to the execution probability of the corresponding header block. The execution probability of normal blocks can be calculated according to the mathematical expectation of the header block. And the mathematical expectation of the header block conforms that the time of continuous states staying obeys geometric distributing. When it is assumed that entering a loop means the failure of a geometric distributing and exiting a loop means the success of a geometric distributing, the distribution of loop structures is consistent with the geometric distribution. If we assign a certain probability to a Bernoulli experiment, the mathematical expectation will be calculated in the geometric distribution. Therefore, calculating the mathematical expectation of a loop structure relies on the success probability of loop iterations. Since the success probability of a Bernoulli experiment for a loop is the same as the execution probability of the corresponding header block, the mathematical expectation of a loop is consistent with the expectation of the header block. In summary, we can calculate the

mathematical expectation of the header block through the success or failure probability of a loop iteration, and then we can infer expectations of all blocks in loops according to the Markov process.

Figure 2 shows the three typical loop structures in the control flow graph of the target application. Figure 2(a) shows that the control flow graph contains a single top-level loop structure. The basic block A is the entry block and the basic block E is the exit block. The loop consists of block B, block C, and block D. We can infer that the execution probability of block B and block D are both 1. Since the jumping probability of the backedge is 0.5, the success probability of corresponding geometric distributing is also 0.5. According to formula (2), the mathematical expectation of the loop is 2. As aforementioned, the mathematical expectation of block C or D is calculated through the Markov process, which indicates that the mathematical expectation is reasonable as static contribution in this case:

$$E(X) = \frac{1}{P(X)}. \quad (2)$$

In addition to the simple single loop structure, there are many nested loop structures in real-world applications. As Figure 2(b) shows, a loop is a subloop of a top-level loop, and we regard the nested loop structure as a nested geometric distribution. The feature of nested loop structures is that the top-level loop will be executed first and to jump into the subloop before exiting the top-level loop. After the subloop completes loop iterations, the execution flow will return to the top-level loop. In other words, for top-level loops, the failure probability of geometric distribution must be 1. Therefore, for the nested loop structure, the crucial factor to calculate mathematical expectation is the failure probability of the subloop. We can easily calculate the expectation of the subloop's header block based on the failure probability of the subloop and then infer probabilities of all blocks according to the Markov process, until the execution flow exits the current subloop. After exiting the subloop, the mathematical distribution of top-level can be computed by the expectation of the corresponding header block. For example, as Figure 2(b) shows, the execution probability of basic block C is 0.5 and the jumping probability of the edge for C to F is 0.5, and we infer the execution probability of basic block F is 0.25. As a result, the failure probability of the subloop is 0.5:

$$E(X) = \frac{1}{P(X)} * P(Y). \quad (3)$$

According to formula (3), the mathematical expectation of the subloop is 1, and the mathematical expectation of corresponding header block C is assigned a value of 1. The jumping probability of the edge from C to D is 0.5, the mathematical expectation of D is 0.5 based on the Markov process. Thus, the probability of returning the top-level loop from the subloop (i.e., the jumping probability of the edge from D to B) is 0.5. Consequently, the success probability of the top-level loop is 0.5. According to formula (2), the mathematical expectation of the top-level loop is 2, which means the mathematical expectation of basic block C is 2. In

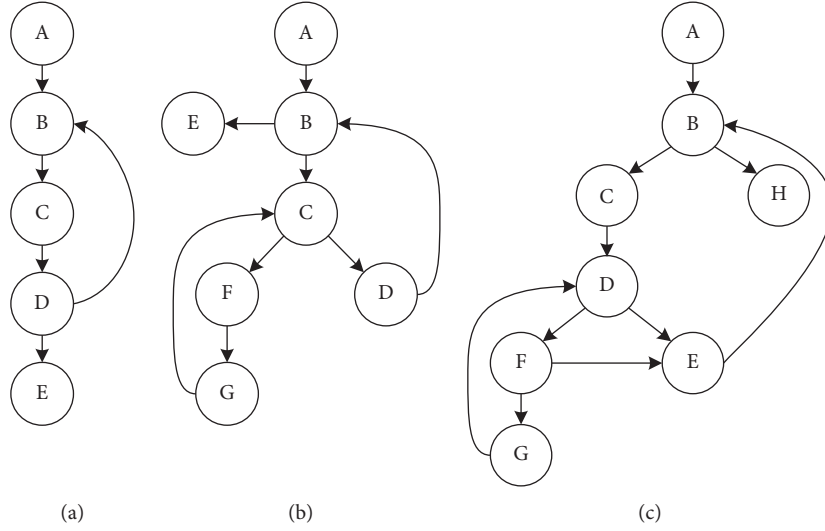


FIGURE 2: Three typical loop structures. (a) The CFG includes a single top-level loop. (b) The CFG includes a nest loop structure. (c) The CFG includes a loop containing a jumping statement.

total, we have calculated that the mathematical expectation of C's successors (block D and block F) is both 1.

Figure 2(c) shows that the loop structure contains jumping statements (for example, break and continue) that may abort the loop iteration and jump out of the loop. In this case, the control flow may not complete an entire loop iteration and exit the loop before the header block, which apparently increases the failure probability of the corresponding Bernoulli trial for the loop. The failure or success probability of this loop structure could be inferred through the execution probability of the header block and then correctly calculate the mathematical expectation of other basic blocks. As Figure 2(c) shows, in the subloop that is composed of basic blocks D, F, and G, a branch from block F to block E jumps out of the subloop, leading to the end of the iteration. The execution probability of block C and D are both 0.5, and the jumping probability of the edge from block D to block F is 0.5; thus, the execution probability of block F is 0.25. According the Markov process, the execution probability of block G is 0.125. As same as the subloop in Figure 2(b), we calculate the success probability of the subloop is 0.75.

As formula (3) shows, the mathematical expectation of the corresponding header block D is 0.67, as the expectation of the subloop is also 0.67. The mathematical expectation of the exit block F is 0.33, and the jumping probability of the edge from F to G is 0.5; thus, the expectation of block G is 0.17, and the expectation of block E is 0.5. The success probability of the top-level loop is 0.5, and then the mathematical expectation of the top-level is 2 based on formula (2). To sum up, we have calculated that the expectation of block D is 1.33 and the expectation of block F is 0.67.

According to the above three loop structures, we can calculate the mathematical expectation of any loop structure as follows:

$$E(\text{Loop}_i) = \frac{E(\text{header}_i)}{E(\text{header}_i) - E(\text{header}_i) * \text{Prob}_{\text{jump}}(\text{backedge}_i)}, \quad (4)$$

where  $E(\text{header}_i)$  is the mathematic expectation of the corresponding header block in  $\text{Loop}_i$ , and the  $\text{Prob}_{\text{jump}}(\text{backedge}_i)$  stands for the jumping probability of the back edge from any exiting block to the header block  $\text{header}_i$ . Based on the expectation of the loop, we infer the expectation of any block in the loop through the Markov process.

As aforementioned, we propose a static contribution calculation approach based on the probability of geometric distribution and mathematical expectation, which leads the complex loop structure that brings about nested control flow jumps. Combined with Markov process, this approach can accurately calculate the contribution of all basic blocks in any control flow graph.

Algorithm 1 demonstrates the process of calculating the mathematical expectation of each block in any loop structure. It first traverses all successors of any visited basic block (line 1), and calculate the mathematical expectations of visited blocks based on the execution probability and the jumping probability of the edge between visited blocks (line 3~5). Then, we correctly process the jumping probability of the backedge as the above three cases and eliminate the visited backedge to convert the loop into a directed acyclic graph (line 7~9). Last, we traverse the rest of visited blocks and add these blocks into the analysis queue (line 11~15).

**3.2. Dynamic Contribution of Basic Block.** We assume that each basic block has the same execution probability when calculating the static contribution. Successors of the same basic block have the same jumping probability. However, in the real-world application, jumping probabilities of different

```

Input:  $B$ : the set of visited basic blocks
       $B$ : the set of visited basic blocks
Output: Calculating the expectation of each basic blocks
(1) if  $\forall b \in B, \exists c \in \text{Succ}(b) \cap c \in B$  then
(2)   if  $\text{jumpweight}(b, c) \geq 0$  then
(3)      $\text{temp} = \text{weight}(c) \div (\text{Weight}(c) - \text{jumpweight}(b, c) \times \text{weight}(c))$ 
(4)     for each block  $\rho \in \text{pred}(\text{succ}(b))$  do
(5)        $\text{jumpweight}(\rho, c) = \text{temp} \times \text{jumpweight}(\rho, c)$ 
(6)     end for
(7)      $\text{jumpweight}(b, c) = -1$ 
(8)     remove  $(B, \text{Succ}(c))$ 
(9)     add  $(U, c)$ 
(10)  else
(11)    add  $(B, b)$ 
(12)  end if
(13) else
(14)  add  $(U, c)$ 
(15)  add  $(B, b)$ 
(16) end if

```

ALGORITHM 1: Calculate expectation of basic blocks in a loop structure.

successors for the same branch are different. For example, in a loop, the jumping count of the branch from the latch block to the header block and the jumping count of the branch from the latch block to the exiting block satisfy the following equation:

$$\frac{C_H}{C_E} = \text{Iter}, \quad (5)$$

where  $C_H$  is the jumping count of the branch from the latch block to the header block, and  $C_E$  is the jumping count of the branch from the latch block to the exiting block; Iter is the number of iterations. If we only assume each successor of any branch has the same jumping probability, the calculated static contribution is inaccurate. Therefore, our approach contains not only static contributions determined by control flow but also dynamic contributions inferred from runtime information. We count the number of executions to determine the triggered frequency of a discovered block, which will be used to evaluate the dynamic contribution.

However, there are two problems with simply counting the number of executions as a dynamic contribution of any block. First, in real-world applications, the sequential execution flow will lead to the nonuniform triggering probabilities of basic blocks. Different basic blocks achieve different functions, and there are fixed executed sequences of blocks for any function. Some basic blocks perform a shallow logic of the application, which may be executed frequently. Some other basic blocks complete a deep logic of the application and will only be triggered if all conditions in the path are satisfied. In our intuition, a basic block in deeper logic has a greater dynamic contribution because it is more difficult to trigger. Second, the number of iterations for the fuzzing process is numerous. There will be astonishing runtime overhead if we count the number of triggers for each block. Meanwhile, the number of discovered blocks will continue to be added as the fuzzing process is iterated. We

have to traverse all discovered blocks in each iteration to calculate statistics, which causes more and more serious performance overhead. In summary, we consider the negative impacts caused by the above two problems when calculating the dynamic contribution. For the first problem, basic blocks in shallow logic will be executed more frequently than basic blocks in deep logic, and we utilize the discovery order of basic blocks to solve it. For the second problem, the more and more serious performance overhead as iterating, we set a threshold  $T$  of iterations to mitigate the performance overhead. We only record executions of discovered blocks within  $T$  iterations to evaluate the dynamic contribution of the discovered block.

If the number of fuzzing iterations is less than the threshold  $T$ , we accurately calculate the dynamic runtime probability based on the statistical data of each basic block as follows:

$$P_{\text{dyn}}(\text{block}_i) = \begin{cases} \frac{P_{\text{dyn}}(\text{block}_i) * x}{x + 1}, & (\text{block}_i \text{ non\_executed}), \\ \frac{P_{\text{dyn}}(\text{block}_i) * x + 1}{x + 1}, & (\text{block}_i \text{ executed}), \end{cases} \quad (6)$$

where  $P_{\text{dyn}}(\text{block}_i)$  is the dynamic runtime probability of basic block  $i$  and the argument  $x$  stands for the number of the iterations.

If the number of fuzzing iterations is bigger than the threshold  $T$ , we simplify the calculation of the dynamic runtime probability as follows:

$$P_{\text{dyn}}(\text{block}_i) = \begin{cases} \frac{P_{\text{dyn}}(\text{block}_i) * (T - 1)}{T}, & (\text{block}_i \text{ non\_executed}), \\ \frac{P_{\text{dyn}}(\text{block}_i) * (T - 1)}{T}, & (\text{block}_i \text{ executed}). \end{cases} \quad (7)$$



Algorithm 2 shows how to calculate the dynamic contributions of basic blocks as the above approach. As aforementioned, there are two situations which are divided by the number of the fuzzing iterations (line 2 and line 8). In first case, we calculate the dynamic contribution of each block according to formula (6) (line 3~7). In second case, if the number of the fuzzing iterations is greater than the threshold  $T$ , we attempt to infer the dynamic contribution based on formula (7) (line 9~13).

## 4. Design and Implementation

In the previous section, we proposed an approach to infer the static and dynamic contributions of basic blocks through the control flow of the target application and the runtime information of fuzzing iteration. However, the contribution information cannot guide the fuzzing process to explore more interesting code regions, because the fuzzing process does not utilize contribution information in fuzzing iterations. Therefore, we design a contribution-aware fuzzer named StFuzzer and introduce the main work flow and the crucial implementation details of StFuzzer.

*4.1. Design of StFuzzer.* As aforementioned, we provide a contribution metric based on the control flow and runtime information, which quantifies the contribution of basic blocks for exploring applications. The approach only utilizes the control flow information of target applications and does not use sophisticated data flow analysis. So, this approach can be used for gray-box fuzzing and white-box fuzzing. In white-box fuzzing, the source code can be accessed, and we can accurately obtain the control flow information at compile-time to calculate the static contribution. In gray-box fuzzing, the source code cannot be accessed, and we decompile the target binary to obtain the control flow graph.

Figure 3 shows the main work flow of StFuzzer:

- (1) If the target is a gray-box program, StFuzzer decompiles the target application.
- (2) StFuzzer instruments the target application based on different instrumentation approaches according to whether the target application is gray-box or white-box.
- (3) StFuzzer calculates the static and dynamic contribution of each basic block.
- (4) After the calculation of static and dynamic contributions, the fuzzing process prioritizes favorable seeds based on seed selection strategy, which are mutated to generate fresh test cases.
- (5) Executing new test cases in the target application, and updating test cases that trigger new coverage, and returning the execution information for the calculation of dynamic contributions.
- (6) Go to 3.

Since static contributions of the target application are fixed, the static contribution calculation is only performed

once when the fuzzing process accesses a target application for the first time. After the static contribution is calculated, the static contribution information of basic blocks will be stored in a map file in the form of <key, value>, where key means the ID of the basic block and the static contribution information as values to calculate the relative information. In the next iteration of StFuzzer, the fuzzing process will read the static contribution information from the map file into in-process memory. So, the calculation of static contributions does not increase the runtime performance overhead. StFuzzer performs dry run of all test cases to collect and calculate the dynamic contribution information and initializes all test cases in the seed queue. Then, we execute the test case in the target application and count the number of executions for each basic block. The number of executions is the basic information for calculating dynamic contributions of basic blocks. It is worth noting that the fuzzing process would affect dynamic contributions of different blocks, and the dynamic contribution information also has a crucial impact on the program exploration of the fuzzing process.

The state-of-art coverage-guided fuzzer AFL believes that the importance of all paths is the same. However, in the real-world application, some execution paths would trigger the core functions of the application, and some paths only call error handling function to end the execution. Obviously, the former plays a more important role in the fuzzing process. In our work, the importance of paths depends on the contributions of corresponding basic blocks. In order to implement the feedback loop between dynamic contributions and heuristic exploration, the contribution information would be applied to the evolutionary seed selection algorithm. StFuzzer calculates the weight of each test case according to the triggered blocks and corresponding static and dynamic contributions of these blocks and proposes a weighted seed selection algorithm.

The weighted seed selection algorithm is the most important module of StFuzzer, because it achieves the feedback in the fuzzing loop. The weighted seed selection algorithm selects seeds which trigger more basic blocks with higher contributions. We believe mutating these seeds has a greater probability of triggering more blocks with high contributions, which is conducive to improve the efficiency of application exploration and vulnerability discovery. It is worth noting that some basic block may be triggered multiple times by a test case in a fuzzing iteration. So, a block may have an excessive dynamic contribution. To avoid a basic block with the excessive contribution causing the fuzzing process to trap a special region of the target, we consider translating the static and dynamic contribution to the weight of a basic block as follows:

$$W(\text{block}_i) = \frac{\log(1/C_{\text{sta}}(\text{block}_i))}{\log 2} + \frac{\log(1/C_{\text{dyn}}(\text{block}_i))}{\log 2}, \quad (8)$$

where  $C_{\text{sta}}(\text{block}_i)$  is the static contribution of block  $i$ , and  $C_{\text{dyn}}(\text{block}_i)$  is the dynamic contribution of block  $i$ . The higher the contribution of a basic block, the higher the



*4.2.2. Contribution Calculation.* The core component of StFuzzer is the contribution calculation, which includes the static contribution and the dynamic contribution. For gray-box applications, we implemented a common interface to analyze the static contribution information according to the control flow graph which is constructed by the IDA decompiler and our analysis script. For white-box applications, we achieved a llvm module pass to calculate the static contribution information.

*4.2.3. Seed Selection.* Evolutionary fuzzers usually select favorable seed as the initial seed used to further mutations. The seed selection usually relies on a certain seed selection strategy. We implemented a module based on the seed scoring approach introduced in the above section to compute scores of all test cases and to prioritize test cases with higher score. The weighted seed selection algorithm can be used to the block coverage metric and the edge coverage metric. We implemented StFuzzer by the block coverage metric to adapt the black-box application. We also implemented a set of extensible interfaces to help StFuzzer utilizing the edge coverage metric in future improvement.

*4.2.4. Mutation Engine.* Mutation is a critical component for a coverage-guided fuzzer. A good mutation engine mutates the initial seed multiple times to traverse code region as much as possible. We implemented an extensible interface to allow StFuzzer to use other mutation engines quickly and support other mutation optimizations such as taint or symbolic execution.

*4.2.5. Crucial Data Manager.* StFuzzer has acquired more and more analysis results during the iteration of fuzzing, which provides a feedback to guide the fuzzing process to explore more basic block with higher contributions. The analysis results mainly include the fixed static contribution of each block and growing dynamic contribution of each discovered block. We implemented an extensible mechanism to store values of static contributions in a bitmap (the ID of each basic block as key), similar to the bitmap storing edge information of AFL. Since the dynamic contribution information is changed as iterations of fuzzing, we implemented an in-process model to store and index dynamic contributions in a list-based structure which is traversed fast.

## 5. Evaluation

In this section, we design a series of experiments to evaluate the efficiency and the effectiveness of bug finding of StFuzzer. And we compare StFuzzer with other fuzzers to show the code coverage and crash growth improvements in real-world applications.

*5.1. Experiment Setup.* Baseline fuzzers: we carefully choose several well-known coverage-guided fuzzers as baseline fuzzers, including AFL, QSYM, and honggfuzz. First, AFL is the most popular and most famous coverage-guided fuzzing.

There are many AFL-family fuzzers that have been implemented based on AFL. Second, QSYM is the art-of-state symbolic execution assisted fuzzer, which designed a special symbolic execution engine to satisfy the fuzzing process. So, QSYM usually has higher coverage than other coverage-guided fuzzers. Meanwhile, the execution speed and efficiency of QSYM is better than other symbolic execution assisted fuzzer such as Driller. Third, honggfuzz is the core fuzzing engine of Google's OSS-fuzz project. Honggfuzz relies on the feature of control flow and the feature of data flow to achieve higher coverage than AFL. Therefore, we chose these fuzzers as baseline fuzzers to ensure the improvement of StFuzzer.

Target applications: we deliberated three factors when choosing the target applications, including functionality diversity, code scale, and popularity. The target applications should be from real-world applications running on smart devices to make sure that StFuzzer is able to recognize zero-day vulnerabilities in real-world applications, not just perform on the dataset. Finally, we chose four popular open source applications, including ffjpeg, libredwg, catdoc, and libsolv. They have diverse functionalities, including popular image parser library (ffjpeg), industry designing library (libredwg), office file parser (catdoc), and operation system component (libsolv). These applications are supported by GNU and open source communities and get frequent maintenance. There are different scales of application, including large scales (libredwg, libsolv) and small scales (ffjpeg and libsolv).

Experiment environment: we run these baseline fuzzers with the same configuration that a virtual machine with two Intel CPU @ 2.40 GHz and 8 GB RAM, running on Ubuntu 18.04. In order to keep the same running of all fuzzers, we use the same shell arguments to run each fuzzer during 24 hours. The code coverage and execution speed of each experiment are compared to evaluate the capability of bug finding and performance overhead.

*5.2. The Impact of Threshold  $T$ .* Our dynamic contribution solution proposed the calculation with different stages to mitigate the performance overhead of calculating dynamic contributions. Specifically, StFuzzer calculates the dynamic contribution by formula (6) if the number of the fuzzing iteration is less than threshold  $T$ . We used formula (7) to calculate the dynamic contribution if the number of the fuzzing iteration is more than threshold  $T$ . The value of threshold  $T$  must influence the overhead of StFuzzer and the accuracy of the dynamic contribution. In this section, we assign different values to threshold  $T$  and discuss the variation of StFuzzer with different threshold  $T$ .

Table 1 demonstrates the value of block coverage triggered by StFuzzer in ffjpeg and catdoc. The first column represents the different value of threshold  $T$ . We have listed the max value in five experiments and the average value of coverage in catdoc and ffjpeg. In catdoc and ffjpeg, StFuzzer obtains the highest coverage when the threshold  $T$  is assigned a value of 100,000. For both of catdoc and ffjpeg, StFuzzer gets the lowest coverage when the threshold  $T$  is

TABLE 1: The block coverage triggered by StFuzzer.

Threshold $T$	catdoc		ffjpeg	
	Max	Average	Max	Average
100	316	312	211	205
1,000	290	288	207	203
10,000	306	300	215	204
100,000	339	322	226	211
1,000,000	291	280	213	209

1,000. In a word, the exploration capability of StFuzzer is the best if the threshold  $T$  is assigned a value of 100,000.

Table 2 shows the number of execution paths triggered by StFuzzer with different values of threshold  $T$  in ffjpeg and catdoc. As same as Table 1, the first column represents the different value of threshold  $T$ . In catdoc and ffjpeg, StFuzzer discovered the largest number of execution paths when the threshold  $T$  is assigned a value of 100,000. In catdoc, StFuzzer found the smallest number of execution paths when the threshold  $T$  is 1,000,000. For different situations in ffjpeg, StFuzzer found the smallest number of execution paths when the threshold  $T$  is 100. Comparing with the case of block coverage, the exploration ability of StFuzzer is the best when the threshold  $T$  is 100,000.

Table 3 shows the average speed of StFuzzer with different values of threshold  $T$  in ffjpeg and catdoc. As same as Table 1, the first column represents the different value of threshold  $T$ . The average speed stands for the number of executions in one second. In catdoc, the average speed reaches maximum when the threshold  $T$  is 100,000. By contrast, in ffjpeg, the average speed is the slowest when the threshold  $T$  is 100,000, but the difference in average speed is too small to be ignored.

**5.3. Conclusion.** The capability of bug finding and application exploring will be the best when the threshold  $T$  is set to 100,000. The influence of the threshold  $T$  on execution speed is limited. Therefore, StFuzzer has a balance of bug finding and runtime overhead if the threshold is set to 100,000. As a result, in the following experiment, we set the threshold  $T$  to 100,000 to attain the best performance of StFuzzer.

**5.4. Code Coverage Evaluation.** Code coverage is one of the most important factors to evaluate a coverage-guided fuzzer. In coverage-guided fuzzing, more code coverage means the more code region has been triggered, the greater the probability of triggering hidden bugs. Some work has demonstrated that the probability of bug discovery increases by about 0.92% when the code coverage increased by one percentage. The code coverage of a coverage-guided fuzzer directly indicates the vulnerability recognizing ability of this fuzzer. Thus, we evaluated baseline fuzzers and StFuzzer on 4 real-world applications for 24 hours to record the code coverage of each fuzzer.

The coverage improvement: it is worth noting that AFL-family fuzzers apply the edge coverage metric that causes many hash collisions in the bitmap that saved edge

TABLE 2: The number of execution paths triggered by StFuzzer.

Threshold $T$	catdoc		ffjpeg	
	Max	Average	Max	Average
100	199	190	231	223
1,000	172	168	253	238
10,000	199	186	276	259
100,000	223	208	279	277
1,000,000	166	152	238	231

TABLE 3: The average execution speed of StFuzzer.

Threshold $T$	catdoc		ffjpeg	
	Max	Average	Max	Average
100	334	322	76	72
1,000	635	622	58	55
10,000	632	631	44	42
100,000	670	662	44	41
1,000,000	423	418	46	43

information. Since our approach calculates the contribution for each basic block, StFuzzer intuitively applies the block coverage metric. With the intention of comparing coverage of all fuzzers uniformly, we intended to transform the edge coverage to the block coverage (i.e., the number of discovered basic blocks).

Table 4 shows the average coverage of different fuzzers on the four applications within 24 hours. Honggfuzz performs the best in baseline fuzzers because it applies complex control flow features and data flow features. Comparing with AFL, StFuzzer triggered much more basic blocks. Especially, in the large-scale application, StFuzzer discovered 646.6% more blocks of libredwg and 250.7% more blocks of libsolv. StFuzzer outperforms the second best fuzzer, honggfuzz, in all applications, at least over 4.6%. In large-scale applications (libredwg and libsolv), StFuzzer discovered more blocks than honggfuzz at least over 6.5%.

Figure 4 shows the growth of block coverage over time. The trend of growth indicates that block coverage of AFL and QSYM has been rising for a while, but in a short period of time, it has reached the upper limit of block coverage much lower than honggfuzz and StFuzzer in these target applications. In all target applications, StFuzzer has attained a substantial increase in coverage at the beginning stage of the fuzzing and is usually faster than AFL and QSYM (based on the derivative of the coverage growth curve). In the beginning stage, the coverage growth curve of StFuzzer is similar to that of honggfuzz. However, after this stage, StFuzzer is able to discover more blocks than honggfuzz in the exploration stage. It demonstrates that our contribution-aware optimization has a better effect on program exploration than the optimization applied in honggfuzz.

**5.5. Conclusion.** StFuzzer could discover more blocks than all baseline fuzzers, which demonstrates the vulnerability recognizing ability of StFuzzer is better than that of other fuzzers. The improvement of vulnerability recognizing

TABLE 4: The average coverage triggered by various fuzzers.

Application	AFL	QSYM	honggfuzz	StFuzzer
catdoc	121 (+158.7%)	123 (+154.5%)	283 (+10.6%)	313
ffjpeg	91 (+125.3%)	79 (+159.5%)	196 (+4.6%)	205
libredwg	2334 (+646.6%)	3439 (+406.7%)	16366 (+6.5%)	17425
libsolv	2227 (+250.7%)	2260 (+245.5%)	7245 (+7.8%)	7809

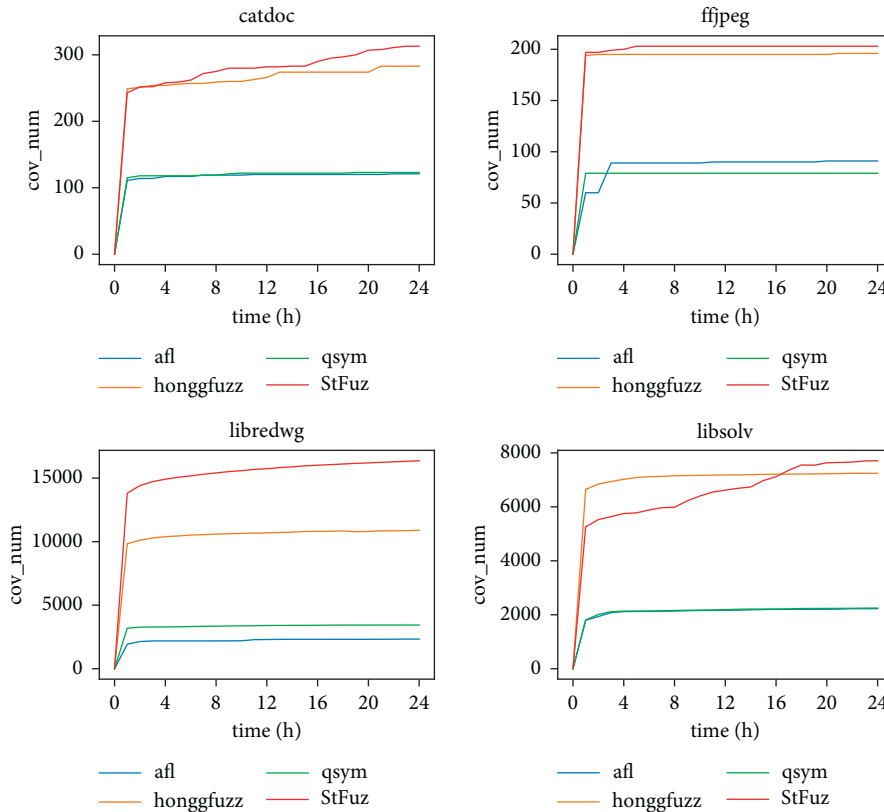


FIGURE 4: The growth of block coverage for each fuzzer.

ability is more obvious in small-scale applications such as catdoc and ffjpeg.

**5.6. Execution Path Evaluation.** Block coverage metric records the triggered blocks, but the execution information of block coverage is not enough accurate. Path coverage metric will track the order of all blocks, providing the most complete code coverage information. To be clearer, we count the number of seeds in the seed queue as path coverage, which is better to show the improvement of our contribution-aware optimization. In the fuzzing iteration, executing a test case will trigger an execution path. The more execution paths, the more triggered execution states of the target application. In most case, the vulnerability is only reproduced in a specific execution path or specific function call sequence. If a test case has triggered a new execution path, the test case will be stored in the seed queue. The more test cases in the seed queue stand for that the more execution states have been explored by the fuzzing process. Therefore, we used the number of test cases in the seed queue to evaluate the execution path triggered by different fuzzers.

Table 5 shows the number of triggered execution paths by different fuzzers. The first column stands for the different target applications. Comparing with AFL and QSYM, StFuzzer finds much more execution paths in these applications. Especially, in libredwg, StFuzzer found 1030.2% more execution paths than AFL. On average, StFuzzer found 15.9% more paths than the second best fuzzer, honggfuzz, found 327.9% more paths than AFL, and found 120.2% more paths than QSYM.

**5.7. Conclusion.** It demonstrated that StFuzzer outperforms all baseline fuzzers in terms of path exploration. StFuzzer can trigger more execution states of a target application within a certain period of time. The improvement of path exploration shows that the contribution-aware approach is beneficial to common coverage-guided fuzzing.

**5.8. Zero-Day Vulnerability Evaluation.** The basic functionality of any fuzzer is finding potential bugs or vulnerabilities in the target application. Therefore, it is the most

TABLE 5: The number of execution paths triggered by various fuzzers.

Application	AFL	QSYM	honggfuzz	StFuzzer
catdoc	80 (+158.8%)	55 (+276.4%)	143 (+44.8%)	207
ffjpeg	145 (+89%)	135 (+103%)	250 (+9.6%)	274
libredwg	1978 (+1030.2%)	13100 (+70.7%)	20932 (+6.8%)	22356
libsolv	8370 (+33.8%)	8578 (+30.5%)	10954 (+2.2%)	11195

direct indicator to reflect the efficiency of a fuzzer that whether the fuzzer finds zero-day vulnerabilities in target applications. We chose the number of zero-day vulnerabilities discovered as the crucial indicator of the last trial to evaluate the effectiveness of contribution-aware optimization for coverage-guided fuzzing.

Usually, a coverage-guided fuzzer will trigger the same bug multiple times during the loop iterating of fuzzing, because the fuzzing process repeatedly mutates a seed that triggered a bug. Duplicated crash samples disturb the exploitable analyzing process of the bug, which is undesirable for any security researchers. Since all target applications of our experiments are open source, we recompiled all target application with AddressSanitizer to filter out crash samples with the same root cause. In total, StFuzzer found 5 zero-day vulnerabilities in these applications. And we have submitted these vulnerabilities to corresponding vendors. These vulnerabilities have been confirmed by vendors and have been fixed in the latest version.

Table 6 shows the number of zero-day vulnerabilities discovered by different fuzzers. StFuzzer found 5 zero-day vulnerabilities, and the number of discovered vulnerabilities is more than that of all baseline fuzzers. It is worth noting that StFuzzer found a vulnerability in libredwg, and other fuzzers could not find this vulnerability. StFuzzer discovered 150% more vulnerabilities than the second best fuzzer, honggfuzz, which indicates that our contribution-aware optimization has a good effect on vulnerabilities recognition.

**5.9. Conclusion.** The contribution-aware optimization proposed in this paper helps the fuzzing process to identify more potential vulnerability. Compared with baseline fuzzers, StFuzzer identified much more zero-day vulnerabilities in these target applications.

**5.10. Performance Overhead Evaluation.** As aforementioned, the calculation of the static contribution for each basic block is performed before the first iteration of the fuzzing process. Calculating the static contribution does not introduce runtime performance overhead of the fuzzing process. However, the calculation of the dynamic contribution for each basic block relies on the runtime execution information of basic blocks discovered, which involves a dynamic calculating procedure. The calculation of the dynamic contribution causes some runtime overhead of the fuzzing process. We evaluated the performance overhead introduced by the additional calculation of the dynamic contribution according to the execution speed of each fuzzer.

Figure 5 shows the average execution speed of each fuzzer in these four applications. Since AFL is the simplest

TABLE 6: The number of zero-day vulnerabilities found by various fuzzers.

Application	AFL	QSYM	honggfuzz	StFuzzer
catdoc	1	0	1	2
ffjpeg	0	0	0	0
libredwg	0	0	0	1
libsolv	0	1	1	2

coverage-guided fuzzing and does not apply any high overhead optimization, we selected the average execution speed of AFL as the baseline in this trial. In this trial, we made an interesting discovery that the contribution optimization produces different performance overheads depending on the scale of the target application. In small-scale applications (catdoc and ffjpeg), StFuzzer has a higher execution effectiveness than AFL, i.e., the average execution speed of StFuzzer is slightly beyond that of AFL. StFuzzer achieved faster execution of each test case in consequence of our contribution-aware optimization guiding the fuzzing process to trigger more basic blocks with higher contributions within a fixed period. In small-scale applications, the performance benefit of the contribution-aware optimization exceeds the runtime overhead of the optimization. In above evaluations, honggfuzz has the best vulnerability recognition ability in the baseline fuzzers, which benefits from the honggfuzz’s optimization that applied control flow features and data flow features. However, due to the enormous performance overhead brought by complex data flow analysis, honggfuzz acquires the worst execution speed. The data flow analysis of honggfuzz generates a lot of performance overhead even in small-scale applications. Comparing with honggfuzz, the contribution-aware optimization of StFuzzer is more useful.

In large-scale applications (libredwg and libsolv), the average execution speed of StFuzzer is slower than that of AFL. It means that the performance overhead for calculating dynamic contributions has increased significantly. In large-scale applications, the core logic of the applications is complex, which leads AFL trapped into the shallow logic and the exploration ended before drilling the core functions. Therefore, AFL and QSYM have a high execution speed in large-scale applications. Comparing with honggfuzz, the average execution speed of StFuzzer is increased by 40% and 116% in libredwg and libsolv, respectively.

**5.11. Conclusion.** The contribution-aware optimization of StFuzzer is a lightweight optimization solution, which introduces less performance overhead. In small-scale applications, since the performance benefit of our optimization,

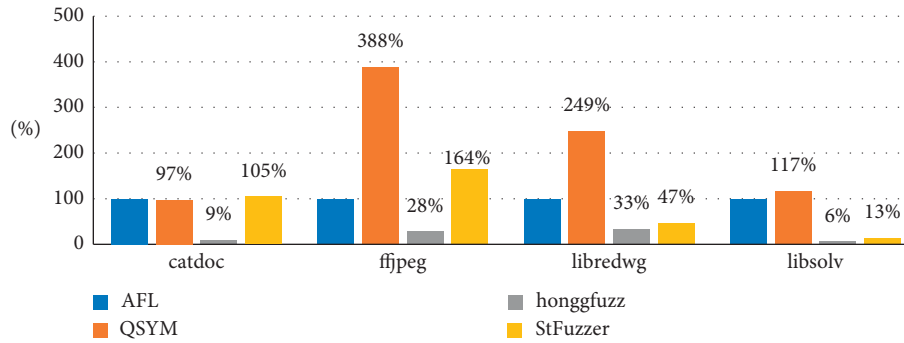


FIGURE 5: The average execution speed of each fuzzer.

StFuzzer has a better performance than AFL. In large-scale applications, the contribution-aware optimization introduces some performance overhead, resulting in a decrease in execution speed. However, the average execution speed of StFuzzer is still much faster than that of honggfuzz.

## 6. Conclusions

In this paper, we propose a contribution-aware optimization solution and implement a novel coverage-guided fuzzer for smart devices, named StFuzzer. We define the contribution of a basic block for the fuzzing process and describe that the basic block with higher contribution will be more useful to device exploration. We show an efficient algorithm to calculate the static contribution of a basic block based on control flow and the approach that evaluate the dynamic contribution through runtime information.

We evaluated StFuzzer with several state-of-the-art coverage-guided fuzzers in four real-world applications running on different smart devices. The result demonstrated that StFuzzer outperforms all baseline fuzzers in terms of vulnerability recognition, application exploration, and code coverage. And we also evaluated the performance overhead of the contribution-aware optimization, which showed that our optimization is lightweight and has good performance in both small-scale applications and large-scale applications.

## Data Availability

The data used in this study are not available.

## Conflicts of Interest

The authors declare no conflicts of interest.

## Acknowledgments

This research was supported in part by the National Natural Science Foundation of China under grant U20B2046, Guangdong Province Key Research and Development Plan under grant 2019B010137004, Guangdong Higher Education Innovation Group 2020KCXTD007 and Guangzhou Higher Education Innovation Group 202032854, and Guangzhou University Graduate Student Innovation Ability Cultivation Funding Program (grant no. 2019GDJC-M16).

## References

- [1] M. Zalewski, "American fuzzy lop," 2013, <http://lcamtuf.coredump.cx/afl/>.
- [2] S. Rawat, V. Jain, A. Kumar, L. Cojocar, C. Giuffrida, and H. Bos, "VUzzer: application-aware evolutionary fuzzing," *National Down Syndrome Society*, vol. 17, pp. 1–14, 2017.
- [3] A. Rebert, S. K. Cha, T. Avgerinos, J. Foote, D. Warren, and G. Grieco, "Optimizing seed selection for fuzzing," in *Proceedings of the 23rd {USENIX} Security Symposium ({USENIX} Security 14)*, pp. 861–875, San Diego, CA, USA, August 2014.
- [4] M. Böhme, V. T. Pham, and A. Roychoudhury, "Coverage-based greybox fuzzing as Markov chain," *IEEE Transactions on Software Engineering*, vol. 45, no. 5, pp. 489–506, 2017.
- [5] O. Aflpin, "Github," 2018, <https://github.com/mothran/aflpin>.
- [6] M. Eddington, *Peach Fuzzing Platform Whitepaper*, p. 34, 2011, <https://www.peach.tech/wp-content/uploads/Peach-Fuzzer-Platform-Whitepaper.pdf>.
- [7] M. Böhme, V. T. Pham, M. D. Nguyen, and A. Roychoudhury, "Directed greybox fuzzing," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp. 2329–2344, Dallas, TX, USA, November 2017.
- [8] S. Gan, C. Zhang, X. Qin et al., "Collafl: path sensitive fuzzing," in *Proceedings of the 2018 IEEE Symposium on Security and Privacy (SP)*, pp. 679–696, IEEE, San Francisco, CA, USA, May 2018.
- [9] T. Yue, P. Wang, Y. Tang et al., "Ecofuzz: adaptive energy-saving greybox fuzzing as a variant of the adversarial multi-armed bandit," in *Proceedings of the 29th {USENIX} Security Symposium ({USENIX} Security 20)*, pp. 2307–2324, Boston, MA, USA, March 2020.
- [10] P. Chen and H. Chen, "Angora: efficient fuzzing by principled search," in *Proceedings of the 2018 IEEE Symposium on Security and Privacy (SP)*, pp. 711–725, IEEE, San Francisco, CA., USA, May 2018.
- [11] H. Chen, Y. Xue, Y. Li et al., "Hawkeye: towards a desired directed grey-box fuzzer," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pp. 2095–2108, Toronto, Canada, October 2018.
- [12] S. Wang, J. Nam, and L. Tan, "QTEP: quality-aware test case prioritization," in *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering*, pp. 523–534, Paderborn, Germany, September 2017.
- [13] W. You, X. Wang, S. Ma et al., "Profuzzer: on-the-fly input type probing for better zero-day vulnerability discovery," in *Proceedings of the 2019 IEEE Symposium on Security and Privacy (SP)*, pp. 769–786, IEEE, Sanfransico, CA, USA, May 2019.

- [14] S. Gan, C. Zhang, and P. Chen, "{GEYONE}: data flow sensitive fuzzing," in *Proceedings of the 29th {USENIX} Security Symposium ({USENIX} Security 20)*, pp. 2577–2594, Boston, MA, USA, March 2020.
- [15] Z. M. Jiang, J. J. Bai, K. Lu, and S. H. Hu, "Fuzzing error handling code using context-sensitive software fault injection," in *Proceedings of the 29th {USENIX} Security Symposium ({USENIX} Security 20)*, pp. 2595–2612, Boston, MA, USA, March 2020.
- [16] U. Kargén and N. Shahmehri, "Turning programs against each other: high coverage fuzz-testing using binary-code mutation and dynamic slicing," in *Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering*, pp. 782–792, Bergamo, Italy, August 2015.
- [17] H. Peng, Y. Shoshitaishvili, and M. Payer, "T-Fuzz: Fuzzing by program transformation," in *Proceedings of the 2018 IEEE Symposium on Security and Privacy (SP)*, pp. 697–710, IEEE, San Francisco, CA, USA, May 2018.
- [18] K.-K. Ma, K. Yit Phang, J. S. Foster, and M. Hicks, "Directed symbolic execution," in *Proceedings of the International Static Analysis Symposium*, pp. 95–111, Springer, Venue, Italy, September 2011.
- [19] N. Stephens, J. Grosen, C. Salls et al., "Driller: augmenting fuzzing through selective symbolic execution," *National Down Syndrome Society*, vol. 16, pp. 1–16, 2016.
- [20] I. Yun, S. Lee, M. Xu, Y. Zhang, and T. Kim, "{QSYM}: a practical concolic execution engine tailored for hybrid fuzzing," in *Proceedings of the 27th {USENIX} Security Symposium ({USENIX} Security 18)*, pp. 745–761, Baltimore, MD, USA, August 2018.
- [21] L. Zhao, Y. Duan, H. Yin, and J. Xuan, "Send hardest problems my way: probabilistic path prioritization for hybrid fuzzing," *National Down Syndrome Society*, San Diego, CA, USA, 2019.
- [22] J. Peng, F. Li, B. Liu, K. Chen, and W. Huo, "1dvul: discovering 1-day vulnerabilities through binary patches," in *Proceedings of the 2019 49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pp. 605–616, IEEE, Portland, OR, USA, June 2019.
- [23] Y. Chen, P. Li, J. Xu et al., "Savior: towards bug-driven hybrid testing," in *Proceedings of the 2020 IEEE Symposium on Security and Privacy (SP)*, pp. 1580–1596, IEEE, San Francisco, CA, USA, May 2020.
- [24] G. Lee, W. Shim, and B. Lee, "Constraint-guided directed greybox fuzzing," in *Proceedings of the 30th {USENIX} Security Symposium ({USENIX} Security 21)*, Vancouver, Canada, August 2021.
- [25] R. Swiecki, "Honggfuzz," 2016, <http://code.google.com/p/honggfuzz>.
- [26] Y. Li, B. Chen, M. Chandramohan, S.-W. Lin, Y. Liu, and A. Tiu, "Steelix: program-state based binary fuzzing," in *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering*, pp. 627–637, ACM, Paderborn, Germany, August 2017.
- [27] P. Godefroid, H. Peleg, and R. Singh, "Learn&fuzz: machine learning for input fuzzing," in *Proceedings of the 2017 32nd IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pp. 50–59, IEEE, Urbana, IL, USA, November 2017.
- [28] S. Y. Kim, S. Lee, I. Yun et al., "Cab-fuzz: practical concolic testing techniques for {COTS} operating systems," in *Proceedings of the 2017 {USENIX} Annual Technical Conference ({USENIX}{ATC} 17)*, pp. 689–701, Santa Clara, CA, USA, July 2017.
- [29] H. S. Han and S. K. Cha, "Imf: inferred model-based fuzzer," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp. 2345–2358, Dallas, TX USA, November 2017.
- [30] W. You, P. Zong, K. Chen et al., "Semfuzz: semantics-based automatic generation of proof-of-concept exploits," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp. 2139–2154, Dallas, TX USA, November 2017.
- [31] Z. Tian, C. Luo, J. Qiu, X. Du, and M. Guizani, "A distributed deep learning system for web attack detection on edge devices," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 3, pp. 1963–1971, 2020.
- [32] C. Luo, Z. Tan, G. Min, J. Gan, W. Shi, and Z. Tian, "A novel web attack detection system for internet of things via ensemble classification," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 8, pp. 5810–5818, 2021.
- [33] M. Shafiq, Z. Tian, A. K. Bashir, X. Du, and M. Guizani, "CorrAUC: a malicious bot-IoT traffic detection method in IoT network using machine-learning techniques," *IEEE Internet of Things Journal*, vol. 8, no. 5, pp. 3242–3254, 2021.
- [34] M. Shafiq, Z. Tian, Y. Sun, X. Du, and M. Guizani, "Selection of effective machine learning algorithm and Bot-IoT attacks traffic identification for internet of things in smart city," *Future Generation Computer Systems*, vol. 107, pp. 433–442, 2020.
- [35] J. Liang, Z. Qin, S. Xiao, L. Ou, and X. Lin, "Efficient and secure decision tree classification for cloud-assisted online diagnosis services," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 4, pp. 1632–1644, 2021.



## Research Article

# Security-Oriented Indoor Robots Tracking: An Object Recognition Viewpoint

Yaoqi Yang <sup>1</sup>, Xianglin Wei <sup>2</sup>, Renhui Xu,<sup>1</sup> Laixian Peng <sup>1</sup>, Yunliang Liao,<sup>1</sup> and Lin Ge<sup>1</sup>

<sup>1</sup>College of Communications Engineering, Army Engineering University of PLA, Nanjing 210000, China

<sup>2</sup>63rd Research Institute, National University of Defense Technology, Nanjing 210007, China

Correspondence should be addressed to Laixian Peng; [lxpeng@hotmail.com](mailto:lxpeng@hotmail.com)

Received 11 July 2021; Revised 5 August 2021; Accepted 17 August 2021; Published 16 September 2021

Academic Editor: Muhammad Ahmad

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

Indoor robots, in particular AI-enhanced robots, are enabling a wide range of beneficial applications. However, great cyber or physical damages could be resulted if the robots' vulnerabilities are exploited for malicious purposes. Therefore, a continuous active tracking of multiple robots' positions is necessary. From the perspective of wireless communication, indoor robots are treated as radio sources. Existing radio tracking methods are sensitive to indoor multipath effects and error-prone with great cost. In this backdrop, this paper presents an indoor radio sources tracking algorithm. Firstly, an RSSI (received signal strength indicator) map is constructed based on the interpolation theory. Secondly, a YOLO v3 (You Only Look Once Version 3) detector is applied on the map to identify and locate multiple radio sources. Combining a source's locations at different times, we can reconstruct its moving path and track its movement. Experimental results have shown that in the typical parameter settings, our algorithm's average positioning error is lower than 0.39 m, and the average identification precision is larger than 93.18% in case of 6 radio sources.

## 1. Introduction

**1.1. Motivation and Background.** Indoor robots are becoming increasingly popular in the market in view of their beneficial applications, ranging from navigating and sweeping to healthy-caring. In combination with artificial intelligence (AI), they are anticipated to drastically change people's daily lives. Although these with advantages, indoor robots still face severe cyber and physical threats due to their hardware and software vulnerabilities [1]. In extreme cases, even the simplest cleaning robots could be manipulated to launch aggressive physical attacks, such as assassinating revealed in [2] and eavesdropping on private conversations in [3–9]. Therefore, it is essential to track the nonstationary robots for security besides traditional efficiency concerns.

By literature review, we identify three types of alternative techniques that could be utilized for indoor robots' tracking. In the first place, a robot could be tracked using indoor navigation technique, where multiple anchors that

transmit signals are needed for the robot to derive its own position based on its received signals. Note that to perform the tracking task, we need the robot to report its position [10]. This process could be easily spoofed and spoiled by selfish or malicious robots. In the second place, based on SLAM- (simultaneous localization and mapping-) based technique, a robot may obtain and report its position through diverse sensors [11]. Obviously, this is only feasible for the 'honest' robots. In the third place, image recognition or video recognition could be adopted to identify and distinguish different robots if cameras could be installed on the ceiling [12, 13]. However, this alternative is very sensitive to light condition and obstacles. In summary, existing efforts could not fulfill the requirements for malfunctioned robots tracking task due to two challenges. On one hand, they usually require the cooperation from hijacked or malicious robots. On the other hand, they are sensitive to indoor multipath effects (as for radio-based techniques) or illumination conditions (as for image-based solutions).

*1.2. Related Work.* For ease of presentation, the state of art efforts in indoor localization is shown in Table 1. As shown in Table 1, the efforts in the indoor localization are classified by their method, environment settings, and the one-time object identification consideration. It can be seen that most methods do not take the dynamic environment into the environment, while some traits in the dynamic environment would change, such as wireless channel. Besides, Table 1 is self-explainable, and no existing efforts consider the identification issue for the localized objects in one time. In order to make up this research gap, a one-time indoor object localization and identification scheme is proposed in this paper.

*1.3. Main Work and Contribution.* In this backdrop, this paper aims to develop an active but nonintrusive indoor robots tracking algorithm. Our tracking process contains two steps, i.e., identification step and localization step, and each indoor robot is treated as a wireless radio source. In the identification step, a YOLO v3 network [16] is deployed to identify and distinguish different robots' signals based on the constructed RSSI maps. Then, each robot's position is derived based on the identification results in the localization step. Our main contributions are threefold:

- (1) An RSSI map construction scheme is proposed in the identification step, which has a satisfactory localization resolution with low cost by deploying a small number of monitors.
- (2) After establishing RSSI maps, a network based on YOLO v3 is trained and applied for multiple robots recognition; then, the bounding boxes information of YOLO v3 [16] is further utilized on the recognition results to refine the localization accuracy. To our best knowledge, this is the first paper that treats indoor robot tracking from the viewpoint of object recognition.
- (3) A series of experiments are conducted on a collected dataset to evaluate effectiveness of our algorithm. Results have shown that in typical indoor environments, our proposal's positioning error is less than 0.39m with a recognition precision higher than 93.18% in case of 6 radio sources.

The remaining part of this paper is organized as follows. Section 2 introduces the system model. In Section 3, identification and localization algorithm for indoor radio sources is detailed. Evaluation experiments are conducted to verify the correctness and effectiveness of the proposed algorithm in Section 4. Finally, a brief conclusion is provided in Section 5.

## 2. System Model

The scenario we considered here is that several indoor robots work on a floor, named the task area, of a building; and there are several classrooms, meeting rooms, and corridors on this floor. To realize efficient control and navigation, a robot usually has a communication module and could be treated as an

indoor radio source. Therefore, we could track different robots if we can recognize and distinguish their equipped radios. For the ease of description, each radio source refers to a robot in the following analysis.

*2.1. Task Area Model.* A Cartesian coordinate system is established for the task area to assist radio source localization or tracking, in which an origin, two perpendicular axes, and their positive directions are determined based on the floor plane. In our setting, the origin locates at the northeast corner of the floor, while east and north directions are the positive directions of the horizontal and vertical axes, respectively. The set of to-be-tracked radio sources, i.e., indoor robots, is  $\mathcal{S} = \{S_1, S_2, \dots, S_N\}$ , where  $S_i$  denotes the  $i$ -th radio source, and  $N$  is the number of radio sources. All radio sources' positions at time  $t$  are  $\mathcal{L} = \{(x_1(t), y_1(t)), (x_2(t), y_2(t)), \dots, (x_N(t), y_N(t))\}$ , where  $x_i(t)$  and  $y_i(t)$  are the  $i$ -th radio source's abscissa and ordinate, respectively.

A number of monitors are deployed for tracking purpose through collecting the RSSI values. To determine their respective positions, the task area is divided into  $(m-1) \times (n-1)$  rectangular areas with the same size, and the monitor is placed at each vertex of all rectangles. In this way, a total of  $m \times n$  monitors are placed. Denote the set of monitors as  $\mathcal{D} = \{D_{1,1}, D_{1,2}, \dots, D_{m,n}\}$ , and  $D_{i,j}$  refers to the monitor deployed in the  $i$ -th column and the  $j$ -th row. Each monitor periodically collects its received RSSI value.  $D_{i,j}$ 's monitored RSSI value at time  $t$  is denoted as  $Z(t)_{i,j}$ . Then, all monitors' collected RSSI values at time  $t$  (i.e., 1 of Figure 1(a)) could be recorded as

$$\mathfrak{R}_1(t) = [X|Y|Z(t)], \quad (1)$$

where  $X$ ,  $Y$ , and  $Z(t)$  are all matrices with  $m$  rows and  $n$  columns,  $X_{i,j}$  and  $Y_{i,j}$  are  $D_{i,j}$ 's abscissa and ordinate, respectively, and  $Z(t)_{i,j}$  represents the maximum RSSI value from the collected  $N$  RSSI values by  $D_{i,j}$  at time  $t$ ,  $1 \leq i \leq m$  and  $1 \leq j \leq n$ .

*2.2. Radio Propagation Model.* It is a difficult and time-consuming task to build the radio propagation model for the task area due to the challenges brought by multipath effects. Moreover, the change of the indoor environment may easily bias the established empirical model [14]. To solve this tricky problem, based on the collected RSSI values at the monitors without building any analytical model, an RSSI map is built to extract the radiation trait, which is the spatial distribution of the power within the environment [13] (i.e., the image selected by the bounding box in Figure 1(c)). To be specific, the collected raw dataset  $\mathfrak{R}_1(t)$  could be transformed to an RSSI map, in which an RSSI value is represented by a rectangular shape, as 1–2 in Figure 1(a). Let the set of extracted radiation traits be  $\mathcal{P} = \{P_1, P_2, \dots, P_N\}$ , where  $P_i$  is the  $i$ -th radio source's radiation trait.

*2.3. Tracking Model.* To track multiple radio sources or robots, we have to continuously derive their respective positions at different time snaps. Therefore, in each time

TABLE 1: A comparison of efforts of indoor localization.

Reference	Method	Environment	One-time identification
Liang et al. [12]	Multifingerprint	Time-varying	No consideration
Rahman et al. [13]	Pedestrian dead reckoning	Static	No consideration
Yu et al. [14]	Maximum likelihood estimation	Static	No consideration
Poulose and Han [15]	Bayesian decision theory	Static	No consideration
Zhu et al. [16]	Augmented reality	Static	No consideration
Fazelinia et al. [17]	Fingerprinting with AM signal	Static	No consideration
This work	YOLO v3 approach	Time-varying	Yes

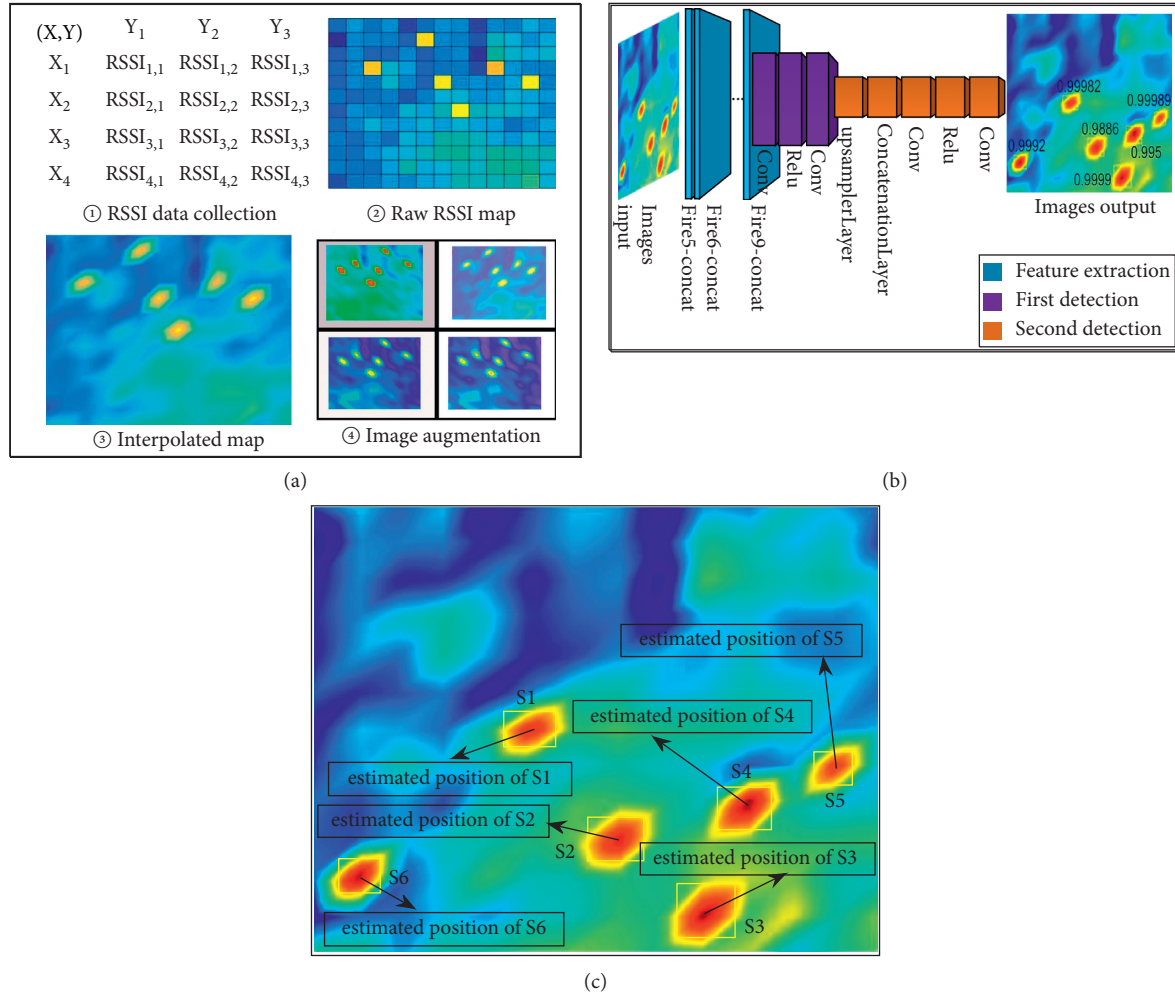


FIGURE 1: The flowchart of the YOLO v3-based radio source tracking algorithm. (a) Images preparation. (b) Radio sources recognition. (c) Radio sources localization.

snap  $t$ , we have to firstly recognize different radio sources based on the established RSSI map and  $\mathcal{P}$  and then derive each radio source's position.

For the recognition purpose, we need to derive a bounding box for each radio source through treating the RSSI map as an image. Then,  $N$  bounding boxes could be obtained for  $N$  radio sources; the set of bounding boxes is denoted as  $\varepsilon = \{E_1, E_2, \dots, E_N\}$ , where  $E_i$  represents the bounding box for the  $i$ -th radio source and contains four elements: the

abscissa and ordinates of the bounding box's top left vertex and the length and the width of the bounding box in order.

The next move is to get all recognized radio sources' positions based on  $\varepsilon$  in combination with the coordinate system embedded in the RSSI map and thus its derived image. Let  $\Lambda = \{(\hat{x}_1(t), \hat{y}_1(t)), (\hat{x}_2(t), \hat{y}_2(t)), \dots, (\hat{x}_N(t), \hat{y}_N(t))\}$  be the set of  $N$  radio sources' estimated positions, where  $\hat{x}_i(t)$  and  $\hat{y}_i(t)$  are the abscissa and ordinate of the  $i$ -th radio source at time  $t$ .

**2.4. Problem Formulation.** To facilitate the problem formulation, we define  $\Omega_{\mathfrak{R}_1(t)}$  as the identification function which can derive  $\mathcal{S}$  based on  $\mathfrak{R}_1(t)$ ;  $\Psi_{\mathfrak{R}_1(t)}$  as the bounding function to derive  $\varepsilon$  based on  $\mathfrak{R}_1(t)$ ; and  $\Theta_\varepsilon$  as the localization function to get the estimated positions  $\Lambda$  based on  $\varepsilon$ :

$$\Omega_{\mathfrak{R}_1(t)}: \{(\mathfrak{R}_1(t)) \longrightarrow (\mathcal{S})\}, \quad (2)$$

$$\Psi_{\mathfrak{R}_1(t)}: \{(\mathfrak{R}_1(t)) \longrightarrow (\varepsilon)\}, \quad (3)$$

$$\Theta_\varepsilon: \{\varepsilon \longrightarrow \Lambda\}. \quad (4)$$

When  $N$  radio sources exist in the task area, and  $M$  radios are identified successfully from the  $N$  radio sources at time  $t$ , this paper aims to jointly minimize the identification error in the identification stage (stage 1) and the positioning error in the localization stage (stage 2):

$$\begin{cases} \min |M - N| & \text{(stage 1)} \\ \min \left\{ \sum_{j=1}^M \left[ (x_j(t) - \hat{x}_j(t))^2 + (y_j(t) - \hat{y}_j(t))^2 \right] \right\} & \text{(stage 2)} \end{cases} \quad (5)$$

### 3. YOLO v3-Based Radio Source Tracking

**3.1. Basic Idea.** After obtaining the interpolated RSSI map and its derived image at time  $t$ , we have to first identify each radio source and determine its bounding box. From the perspective of object recognition in an image, the recognition task could be completed if we can capture the characteristics of each radio source's radiation trait, which represents a radio source's radiation range and intensity in an RSSI map [17]. The process of capturing a radio source's radiation trait equals to deriving  $\Omega_{\mathfrak{R}_1(t)}$  and  $\Psi_{\mathfrak{R}_1(t)}$ . However, there are many ways to solve (2) and (3). For example, the empirical model for indoor object detection and localization based on RSSI is widely utilized because of its simplicity and low-cost. But its positioning accuracy is unsatisfactory since the measured RSSI values are sensitive to the indoor multipath effects. Moreover, it is impractical to distinguish different radio sources based solely on RSSI data.

In this backdrop, this paper adopts deep neural networks for tackling image segmentation, i.e., radio source recognition, in view of their great success in image recognition areas. To be specific, YOLO v3 is adopted for this purpose due to its unique capability to capture the different radio sources' radiation traits and their differences [16]. We could identify each radio source and obtain its bounding box at the same time after conducting YOLO v3 on the image of an RSSI map. In other words, YOLO v3 is adopted to solve both (2) and (3) simultaneously. Compared with empirical model-based solutions, deep neural networks-based methods are robust to indoor multipath effects.

To tackle the localization problem shown in (4), a straightforward idea is treating the center of each radio source's bounding box as its position. However, this will

introduce extra localization errors if the bounding box derived by YOLO is biased. Taking this into consideration, the position of the pixel point with the largest RSSI value within the  $i$ -th radio source's bounding box  $E_i$  is chosen as the radio source's location. If there are multiple pixel points with the same maximum RSSI value, the center position is adopted as the localization result.

**3.2. Indoor Radio Sources Identification and Localization.** As shown in Figure 1, our indoor radio sources tracking method contains three steps: images preparation, radio sources recognition, and radio sources localization. In the first step, raw RSSI values at the monitors are collected to build RSSI maps, which will further be transformed into images using the interpolation theory. Then, a YOLO v3 detector is trained offline on the images for identifying and distinguishing different radio sources, and the trained YOLO v3 network is utilized for online radio sources' recognition. Finally, all radio sources' positions are determined based on their respective bounding boxes and the largest RSSI values of the pixels within the boxes.

**3.2.1. Images Preparation.** Adapting  $\mathfrak{R}_1(t)$  directly to construct the RSSI map will result in two defects. On one hand, the positioning granularity of the RSSI map is determined by the monitors' deployment density. Sparsely deployed monitors will lead to low tracking accuracy while high-density deployment would introduce high deployment cost. On the other hand, biased or even error monitored data are common due to the impacts of malfunctioned monitors or indoor multipath effect. Therefore, to achieve low-cost monitoring while promoting the tracking accuracy, the 2-th Bernstein Bezier interpolation theory [18] is utilized to refine the raw RSSI map, and  $\mathfrak{R}_1(t)$  is expanded to  $\mathfrak{R}_2(t)$ .

As shown in Figure 2,  $D_{i,j}$ ,  $D_{i,j+1}$ ,  $D_{i+1,j}$ , and  $D_{i+1,j+1}$  are four deployed monitors, and their positions are  $(x_{i,j}(t), y_{i,j}(t))$ ,  $(x_{i,j+1}(t), y_{i,j+1}(t))$ ,  $(x_{i+1,j}(t), y_{i+1,j}(t))$ , and  $(x_{i+1,j+1}(t), y_{i+1,j+1}(t))$ , respectively.  $I_1$  and  $I_2$  are the center of gravity of the triangle  $\triangle D_{i,j}D_{i,j+1}D_{i+1,j}$  and triangle  $\triangle D_{i+1,j}D_{i+1,j+1}D_{i+1,j}$ , and their positions are  $(x_{I_1}(t), y_{I_1}(t))$  and  $(x_{I_2}(t), y_{I_2}(t))$ . The purpose of applying 2-th Bernstein Bezier interpolation theory [14] is to derive the RSSI values at  $I_1$  and  $I_2$  without deploying extra monitors.

Next, according to the 2-th Bernstein Bezier polynomial theory, we have the following interpolation formula:

$$z_{I_h} = \sum_{\substack{0 \leq f+g \leq 2 \\ f+g=2}} \frac{2!}{f!g!} b_{i,j} x_{I_h}^f y_{I_h}^g \quad (h = \{1, 2\}). \quad (6)$$

Here,  $z_{I_h}$  is the RSSI value of the interpolation point;  $x_{I_h}$  and  $y_{I_h}$  are the interpolated point's abscissa and ordinate; and  $b_{i,j}$  is called the Bezier ordinates of  $z_{I_h}$  [14]. Let the dataset in the task area after applying the 2-th Bernstein Bezier interpolation be  $\mathfrak{R}_2(t)$ :

$$\mathfrak{R}_2(t) = [X'|Y'|Z'(t)]_{p \times 3q}, \quad (7)$$

where  $X'$ ,  $Y'$ , and  $Z'(t)$  are all matrices with  $p$  rows and  $q$  columns, and they contain the abscissas, ordinates, and RSSI values of the points after interpolation, respectively.  $b_{i,j}$  is called the Bezier ordinates of  $z_{I_h}$  and needs to be determined.

Then,  $\mathfrak{R}_2(t)$  is adopted to establish the refined RSSI map (i.e., the interpolated RSSI map), as 1–3 of Figure 1(a) shows. Finally, image augmentation method presented in reference [19] is applied to expand the number of images in the training dataset, as step 4 in Figure 1(a) shows.

**3.2.2. Radio Sources Recognition.** As shown in the radio sources recognition step in Figure 1, a YOLO v3 network contains three parts: feature extraction network, first detection head, and second detection head, in which 70 net layers consist the neutral networking, and 78 connection tables and 58 learnable tables are used to connect different layers [16].

In the offline training stage, a number of images obtained in the images preparation step are adopted as the training dataset, and each image is labelled with all the radio sources' class and bounding box information. Then, the outputs of the network are the images where the radio sources are recognized with their respective bounding boxes. To refine the weights in the network, back propagation method is adopted and cross entropy is the loss function [16]. The learning rate, the number of training epochs, the number of warm-up periods, and the regularization are set to be 0.001, 3500, 1000, and 0.0005, respectively. In the online recognition stage, the penalty, the confidence, and the overlap thresholds are all set to be 0.5.

**3.2.3. Radio Sources Localization.** To tackle the potential error introduced by biased bounding box derived by YOLO v3 network, the position of the pixel with the largest RSSI values within a recognized radio source's bounding box is chosen as the source's location, as shown in the radio source localization step shown in Figure 1. Therefore, for an identified radio source, (4) can be converted to

$$(\hat{x}_i(t), \hat{y}_i(t)) = \frac{1}{w} \left( \sum_{k=1}^w x_k(t) \sum_{k=1}^w y_k(t) \right) \quad w = 1, 2, \dots, \quad (8)$$

where  $(x_k(t), y_k(t)) \in E_k$ ,  $z_k(t) = \max\{Z(t)\}$ , and  $w$  is the number of the selected pixels in the  $i$ -th bounding box.

**3.2.4. Algorithm Description.** The indoor radio sources tracking algorithm is illustrated in Algorithm 1. Step 1 determines the dataset to construct the RSSI map after interpolation. In steps 2 and 3, the YOLO v3 network is trained, next deriving the image to be recognized in step 5, and the trained YOLO v3 detector is utilized to obtain the identification and localization results in steps 6–9.

## 4. Simulation Results

**4.1. Experimental Settings.** All experiments are conducted in Room 701, Communication Hall, Army Engineering University. The floor plan of the room is shown in Figure 3; the size of room is  $11.2 \text{ m} \times 10.4 \text{ m}$ ; and the vertical and horizontal distances between two neighboring monitors are both 0.8 m [17]. The size of established RSSI map is  $700 \times 525$  pixels and the output image's size is normalized as  $227 \times 227$  pixels to accelerate the training process. In addition, the software for collecting RSSI data is WiFi NetSpot. The tracking period  $T$  lasts for 30 minutes, and it is divided into a number of time intervals with each interval  $t_0$  being 30 seconds. 6 radio sources are investigated, i.e., MECHERVO (S1), Huawei MatePad (S2), Thinkpad T580 (S3), Xiaomi mix2 (S4), HUAWEI P40 (S5), and Thinkpad T480 (S6). These portable devices are carried by Turtlebot Robots. We open all the collected raw RSSI values to the research community (<https://github.com/tracking-data/tracking-project/releases/tag/v1.0>).

### 4.2. Performance Metrics

**4.2.1. Average Identification Precision.** Assume  $a$  times of experiments are conducted in total. In the  $j$ -th experiment, the  $i$ -th radio source emerges  $k_3$  times, and it is detected  $k_1$  times while being recognized correctly by Algorithm 1 for  $k_2$  times. Then, the identification rate of the  $i$ -th radio source in the  $j$ -th experiment will be  $V_i(j) = k_2/k_1$  and the recall rate  $R_i(j) = k_2/k_3$  is derived to measure the false alarm performance. Then, after  $a$  times of experiments, the average identification precision  $V_i^{\text{ave}}$  and the average recall rate  $R_i^{\text{ave}}$  of the  $i$ -th radio source will be

$$V_i^{\text{ave}} = \frac{1}{a} \sum_{j=1}^a V_i(j), \quad (9)$$

$$R_i^{\text{ave}} = \frac{1}{a} \sum_{j=1}^a R_i(j).$$

**4.2.2. Average Positioning Error.** For the  $i$ -th identified radio source, its location  $(\hat{x}_i(j), \hat{y}_i(j))$  can be estimated by Algorithm 1 in the  $j$ -th experiment, and its real position  $(x_i(j), y_i(j))$  is known in advance. With time period  $[0, T]$  ( $T = n \times t_0$ ), denote the average positioning error for the  $i$ -th radio source defined as

$$\mu_i = \frac{1}{n} \sum_{j=0}^n \left[ (\hat{x}_i(j) - x_i(j))^2 + (\hat{y}_i(j) - y_i(j))^2 \right]. \quad (10)$$

### 4.3. Results and Analysis

**4.3.1. Radio Sources Recognition Results.** A straightforward presentation of the radio source recognition results in one single timeslot is shown in the images output layer in Figure 1(b), where 6 radio sources (ranging from S1 to S6)

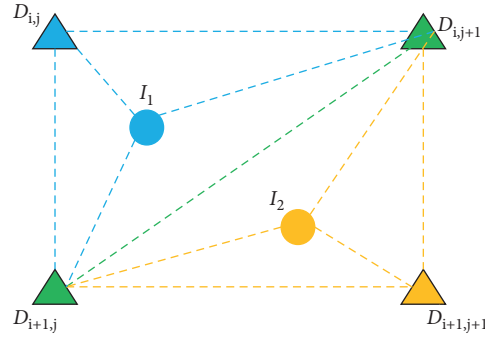


FIGURE 2: The principle of the 2-th Bernstein Bezier interpolation.

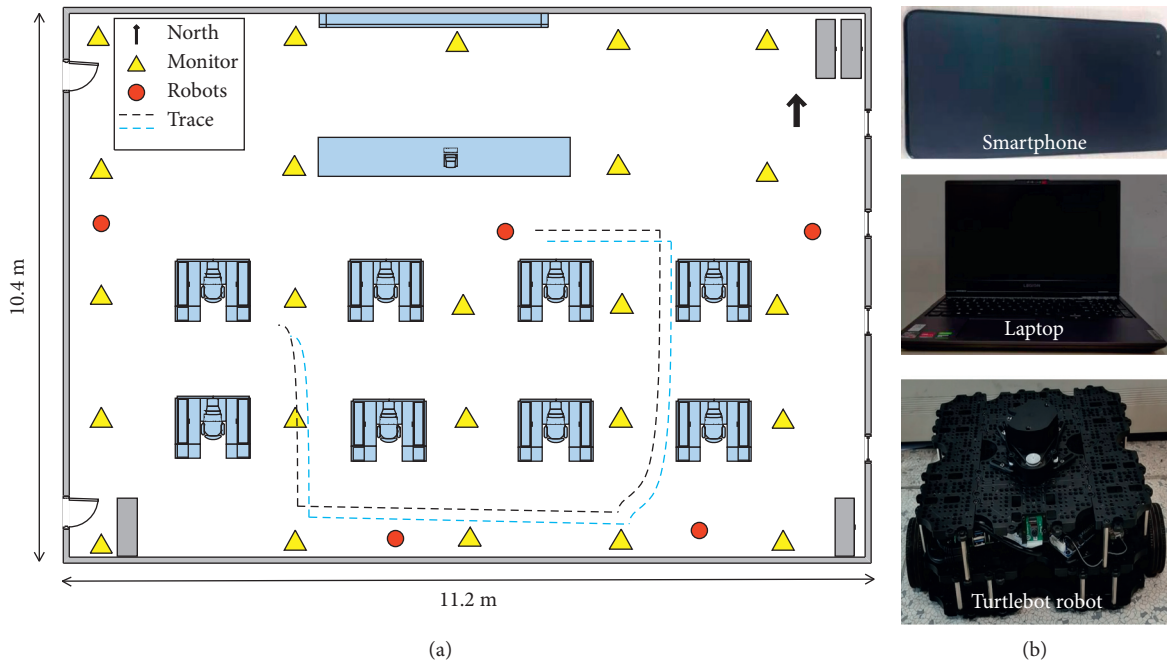


FIGURE 3: The experimental environment. (a) Floor plan. (b) Robot.

are identified with 0.9986, 0.9999, 0.995, 0.9999, 0.9998, and 0.9999 confidence scores, respectively.

Figure 4 shows the 6 radios' identification precision and localization errors. As shown in Figure 4(a), the average identification precisions of the 6 radios are 0.9648, 0.9562, 0.9469, 0.9318, 0.9404, and 0.9440, respectively. The differences between different radios' identification precision lie in the fact that different radio sources have different radiation traits. Generally speaking, the more obvious, i.e., the larger the transmitting power, a radio's radiation trait, the higher its identification precision; and the radiation traits are subject to lots of factors such as the transmitting power, the usage degree, and the position. Figure 4(b) presents the CDF of 6 radios' average positioning error. As shown in Figure 4(b), each radio's average positioning error is less than 0.39 m with a probability higher than 90%.

Figure 5 shows the 6 radio sources' real and estimated traces in 60 timeslots. As can be seen, the difference between

the real and the estimated traces is less than 0.4 m, which is better than 1.54 m with fingerprinting approach in [20]. Besides, the higher the identification precision of a radio source is, the less its positioning error will be. A video is made based on the tracking results and is made publicly available (<https://github.com/tracking-data/tracking-project/releases/tag/v1.0>).

**4.3.2. Generality of the Trained Network.** To evaluate the generality of the trained network, we have tested the trained network in Room 705 and Room 725 located in the same building with different number of robots from S1 to S4, and the testing results are shown in Table 2. From Table 2, we know that the trained network can still achieve high recognition (>90%) and localization ( $\leq 0.65$ m) accuracy in different rooms. Moreover, it is validated that our algorithm still works independent of the number of emerged radios.

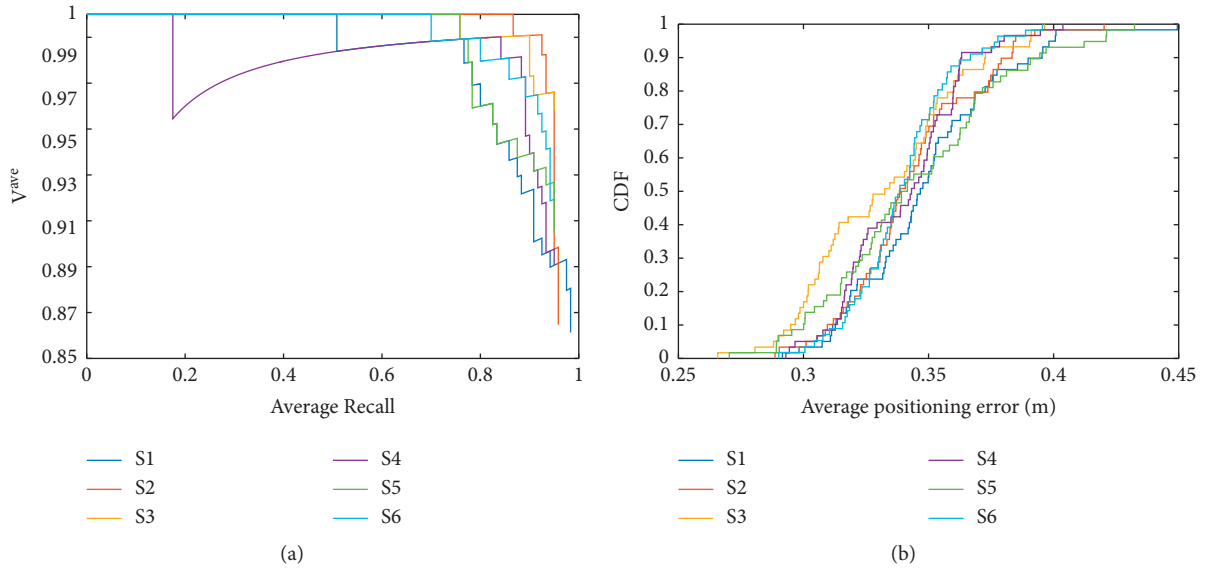


FIGURE 4: The tracking performance of Algorithm 1. (a) Identification performance. (b) Localization performance.

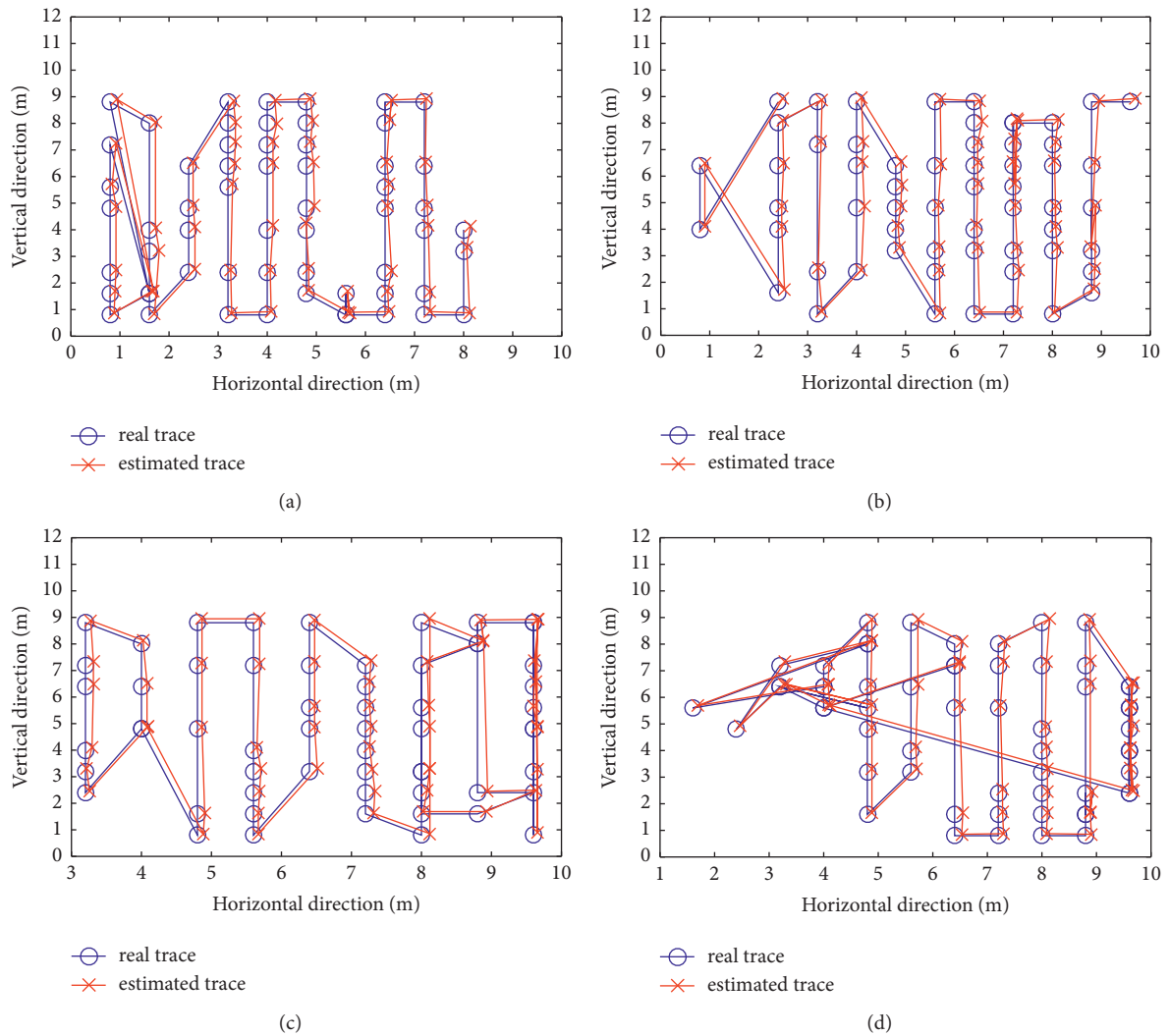


FIGURE 5: Continued.

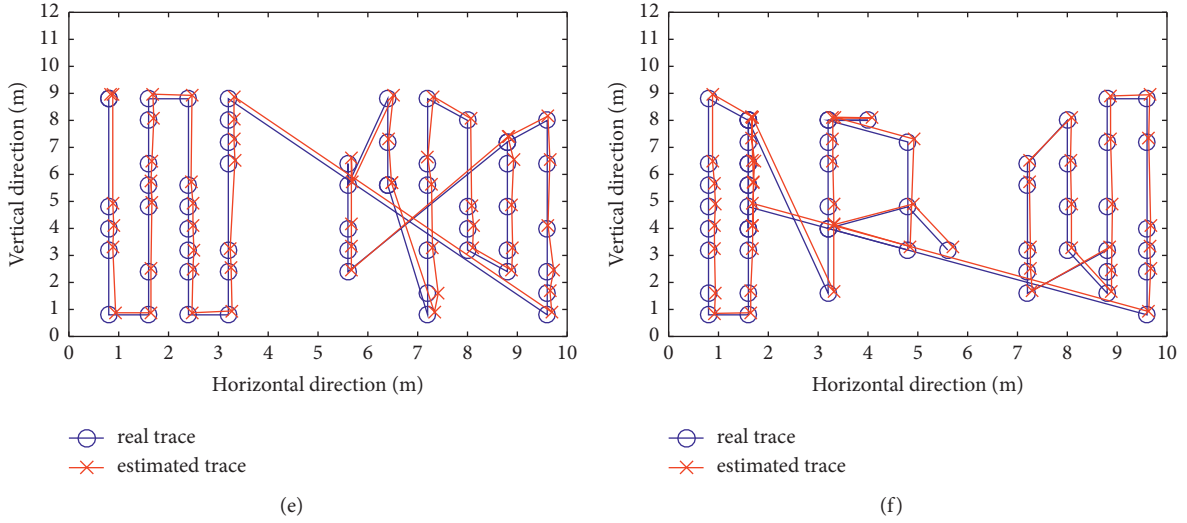


FIGURE 5: The tracking traces of 6 radio sources. (a) The trace of S1. (b) The trace of S2. (c) The trace of S3. (d) The trace of S4. (e) The trace of S5. (f) The trace of S6.

**Input:** Data  $\mathfrak{R}_1^T = \{\mathfrak{R}_1(0), \mathfrak{R}_1(t_0), \dots, \mathfrak{R}_1(T)\}$  recorded by deployed monitors every interval  $t_0$  within  $[0, T]$ ,  $T = n \cdot t_0$

**Output:** Identification results  $(S_i, V_i)$ , localization results  $(\hat{x}_i(t), \hat{y}_i(t))$  ( $1 \leq i \leq M, 0 \leq t \leq T$ )

Offline stage

- (1) Derive  $\mathfrak{R}_2^T$  based on  $\mathfrak{R}_1^T$  according to (6) and (7)
- (2) Configure network parameters (i.e., training epochs)
- (3) Determine  $\Omega_{\mathfrak{R}_2}$  and  $\Psi_{\mathfrak{R}_2}$  by training the network
- (4) Return the trained YOLO v3 network

Online stage

- (5) Derive the RSSI map according to  $\mathfrak{R}_2^T$
- (6) Input the established RSSI map to the network
- (7) Derive identification results as  $(S_i, V_i)$  based on step 6
- (8) Derive localization results as  $(\hat{x}_i(t), \hat{y}_i(t))$  based on (8)
- (9) Return  $(S_i, V_i), (\hat{x}_i(t), \hat{y}_i(t))$

ALGORITHM 1: Indoor radio sources tracking algorithm.

TABLE 2: Generality evaluation results.

Experiment scene	Radio source	$V_i^{\text{ave}}$	$\mu_i$ (m)
Room 705	S1	0.9217	0.49
	S2	0.9029	0.63
	S3	0.9311	0.52
	S4	0.9168	0.65
Room 725	S1	0.9065	0.59
	S2	0.9436	0.57
	S3	0.9179	0.61

## 5. Conclusion

In this paper, we proposed an algorithm to identify and localize indoor robots (radio sources) in the real time. Experiments show that the proposed algorithm can not only do well in the indoor radio sources identification with

93.18% average identification precision but also is good at localizing them with 0.39 m average positioning error under typical parameters settings. In the future, it would be more interesting to extend the proposed algorithm to enable an incremental robots tracking with variable number of unknown robots.



## Data Availability

All data are available within this paper.

## Conflicts of Interest

The authors declare that they have no conflicts of interest or personal relationships that could have appeared to influence the work reported in this paper.

## Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. 61671471).

## References

- [1] M. Thomas, "2020 global threat intelligence report," 2020, <https://at.nttdata.com>.
- [2] M. Brundage, S. Avin, J. Clark, and H. Toner, "The malicious use of artificial intelligence: forecasting, prevention, and mitigation," 2018, <https://www.eff.org>.
- [3] S. Sami, D. Yimin, X. T. Sean Rui, R. Nirupam, and H. Jun, "Spying with your robot vacuum cleaner: eavesdropping via lidar sensors," in *Proceedings of the 18th Conference on Embedded Networked Sensor Systems*, pp. 354–367, Virtual Event, Japan, November 2020.
- [4] J. Song, Q. Zhong, W. Wang, C. Su, Z. Tan, and Y. Liu, "FPDP: flexible privacy-preserving data publishing scheme for smart agriculture," *IEEE Sensors Journal*, vol. 21, no. 16, pp. 17430–17438, 2021.
- [5] L. Zhang, M. Peng, W. Wang, Z. Jin, Y. Su, and H. Chen, "Secure and efficient data storage and sharing scheme for blockchain-based mobile-edge computing," *Transactions on Emerging Telecommunications Technologies*, Article ID e4315, 2021.
- [6] W. Wang, H. Huang, L. Zhang, and C. Su, "Secure and efficient mutual authentication protocol for smart grid under blockchain," *Peer-to-Peer Networking and Applications*, vol. 14, pp. 1–13, 2020.
- [7] W. Wang, H. Xu, M. Alazab, T. R. Gadekallu, Z. Han, and C. Su, "Blockchain-based reliable and efficient certificateless signature for IIoT devices," *IEEE Transactions on Industrial Informatics*, vol. 2021, Article ID 3084753, 2021.
- [8] L. Zhang, Z. Zhang, W. Wang, Z. Jin, Y. Su, and H. Chen, "Research on a covert communication model realized by using smart contracts in blockchain environment," *IEEE Systems Journal*, vol. 2021, Article ID 3057333, 2021.
- [9] Y. Zou, L. Zhang, W. Wang, Z. Jin, Y. Su, and H. Chen, "Resource allocation and trust computing for blockchain-enabled edge computing system," *Computers & Security*, vol. 105, Article ID 102249, 2021.
- [10] D. Dardari, P. Closas, and P. M. Djuric, "Indoor tracking: theory, methods, and technologies," *IEEE Transactions on Vehicular Technology*, vol. 64, no. 4, pp. 1263–1278, 2015.
- [11] R. Liu, S. H. Marakkalage, M. Padmal et al., "Collaborative SLAM based on WiFi fingerprint similarity and motion information," *IEEE Internet of Things Journal*, vol. 7, no. 3, pp. 1826–1840, 2020.
- [12] X. Liang, H. Wang, Y.-H. Liu, B. You, Z. Liu, and W. Chen, "Calibration-free image-based trajectory tracking control of mobile robots with an overhead camera," *IEEE Transactions on Automation Science and Engineering*, vol. 17, no. 2, pp. 933–946, 2020.
- [13] M. M. Rahman, V. Moghtadaiee, and A. G. Dempster, "Design of fingerprinting technique for indoor localization using AM radio signals," in *Proceedings of the 2017 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, pp. 1–7, Sapporo, Japan, September 2017.
- [14] L. Yu, Y. W. Leung, X. Chu, and J. K. Y. Ng, "Multi-Fingerprint for wireless localization in time-varying indoor environment," in *Proceedings of the GLOBECOM 2020 - 2020 IEEE Global Communications Conference*, pp. 1–6, Taipei, Taiwan, December 2020.
- [15] A. Poulou and D. S. Han, "Indoor localization using PDR with Wi-Fi weighted path loss algorithm," in *Proceedings of the 2019 International Conference on Information and Communication Technology Convergence (ICTC)*, pp. 689–693, Jeju Island, Korea (South), October 2019.
- [16] J. Zhu, Q. Chen, and J. Zhang, "Localization optimization algorithm of maximum likelihood estimation based on received signal strength," in *Proceedings of the 2017 IEEE 9th International Conference on Communication Software and Networks (ICCSN)*, pp. 830–834, Guangzhou, China, May 2017.
- [17] M. Fazelinia, M. R. Daliri, and S. Ebadollahi, "Wi-Fi RSS-based indoor localization using reduced features second order discriminant function," in *Proceedings of the 2019 27th Iranian Conference on Electrical Engineering (ICEE)*, pp. 921–924, Yazd, Iran, April 2019.
- [18] I. Amidror, "Scattered data interpolation methods for electronic imaging systems: a survey," *Journal of Electronic Imaging*, vol. 11, no. 2, 2002.
- [19] H. Rizk, M. Torki, and M. Youssef, "CellinDeep: robust and accurate cellular-based indoor localization via deep learning," *IEEE Sensors Journal*, vol. 19, no. 6, pp. 2305–2312, 2019.
- [20] R. Ma, Q. Guo, C. Hu, and J. Xue, "An improved WiFi indoor positioning algorithm by weighted fusion," *Sensors*, vol. 15, no. 9, pp. 21824–21843, 2015.

## Research Article

# Acquiring Data Traffic for Sustainable IoT and Smart Devices Using Machine Learning Algorithm

Yi Huang,<sup>1</sup> Shah Nazir ,<sup>2</sup> Xinqiang Ma ,<sup>1</sup> Shiming Kong,<sup>1</sup> and Youyuan Liu<sup>1</sup>

<sup>1</sup>*Institute of Intelligent Computing and Visualization Based on Big Data, Chongqing University of Arts and Sciences, Chongqing, China*

<sup>2</sup>*Department of Computer Science, University of Swabi, Swabi, Pakistan*

Correspondence should be addressed to Shah Nazir; [snsahnzr@gmail.com](mailto:snsahnzr@gmail.com) and Xinqiang Ma; [xinqma@zju.edu.cn](mailto:xinqma@zju.edu.cn)

Received 21 April 2021; Revised 11 May 2021; Accepted 25 May 2021; Published 19 June 2021

Academic Editor: Muhammad Ahmad

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

Billions of devices are connected via the Internet which has produced various challenges and opportunities. The increase in the number of devices connected to the Internet of things (IoT) is nearly beyond imagination. These devices are communicating with each other and facilitating human life. The connection of these devices has provided opening directions for the smart applications which are one of the growing areas of research. Among these opportunities, security and privacy are considered to be one of the major issues for researchers to tackle. Proper security measures can prevent attackers from interrupting the security of IoT network inside the smart city for secure data traffic. Keeping in view the security consideration of data traffic for smart devices and IoT, the proposed study presented machine learning algorithms for securing the data traffic based on a firewall for smart devices and IoT network. The study has used the dataset of “Firewall” for validation purposes. The experimental results of the approach show that the hybrid deep learning model (based on convolution neural network and support vector machine) outperforms than decision1 rules and random forest by generating a recognition rate of 95.5% for the hybrid model, 68.5% for decision rules, and 78.3% accuracy for random forest. The validity of the proposed model is also tested based on other performance metrics such as f score, error rate, recall, and precision. This high accuracy rate and other performance values show the applicability of the proposed hybrid model to secure data traffic purposes in smart devices. This can be used in many research areas of the smart city for security purposes.

## 1. Introduction

The world population is growing with the passage of time. Various smart devices such as sensor, actuator, and many other smart devices are installed and linked for smooth communication, solving the issues and needs of daily life. The increase in the number of devices connected to the IoT is beyond imagination. These devices are communicating with each other and facilitating human life. The connection of these devices has provided openings directions for the smart applications which are one of the growing areas of research [1–3]. Among these opportunities, security and privacy are considered to be one of the major issues for researchers to tackle. Smart communication is the direct need of modern societies. The role of IoT is understandable in the smart

communication of these devices [4–6]. The technology has mainly focused on efficient well-being of humans and with the protection of environment.

The technologies of IoT has unlimited potentials for developing sustainable and smart devices. The information is generally gathered from physical objects and transmitted through communication media for processing. High computation systems are used for processing of the data for producing meaningful insights and needs. These processes can support the administration of city in providing the essential information for maintaining the services in effective way. The applications of IoT are ensuring the delivery of smart services with efficient utilization of resources. Based on the applications of IoT, a gateway is opened for information processing and facilitating automated governance of

smart cities. So, for maintaining the smooth interaction and communication of the devices in the IoT network, security can be considered as the key part of the IoT network for the smart cities [7].

Various approaches have been practiced for communication of security in the smart cities. Ibrahim et al. [8] developed a framework with the help of the programmed sensor by Arduino board and provided the sensory data into the storage of cloud for gaining access to smart-home connected devices. As security and privacy are the key concerns during the process of personal data collection, Witt and Konstantas [9] proposed a framework for ensuring the security and protection of citizen's privacy in the smart city. Jia et al. [10] elaborated the vulnerabilities in the smart home architecture and devised a threat model and then discussed building a semiautomatic vulnerability detection system for detecting vulnerabilities from all sides before releasing the device. The approach was demonstrated through wide-ranging experiments. Talal et al. [11] presented a study with the aim for establishing security solution of IoT-based smart home for monitoring real-time health in the architecture of telemedicine. Various layers were presented. A detailed review analysis on telemedicine was presented with the focus on the server and client sides, showing the other related studies with applications of smart home for IoT. Sharma et al. [12] offered a model for testing the feasibility and performance of the network in the course of link failure and to switch normal environment. Various parameters were considered for assessing the performance of the offered model. The results of the experiments revealed that the model is capable of detecting the mitigate attacks and can be considered for securing the system and ensure security of users.

Proper security measures is the awful need of smart cities which can prevent attackers from interrupting the security of IoT network inside the smart city and the data of devices inside the network traffic will be safe. Keeping in view the security consideration of data traffic for smart cities and IoT, the proposed study achieved the following contributions:

- (i) To present machine learning algorithms for securing the data traffic based on a firewall for smart devices and IoT network
- (ii) To use the dataset of "Firewall" for validation purposes of the proposed study
- (iii) To show the effectiveness of the proposed approach through experiments of the approach
- (iv) The validity of the proposed model is also tested based on other performance metrics such as f score, error rate, recall, and precision
- (v) Accuracy rate and other performance values show the applicability of the proposed hybrid model for secure data traffic purposes in smart devices

The organization of the paper is as follows. Section 2 depicts the related work to the proposed study. Section 3 shows the methodology of the proposed study with the background of existing associated analysis of literature. Section 4 shows the results and discussion of the paper with evaluation measures of the proposed study. The paper is concluded in section 5.

## 2. Related Work

Researchers are trying to come across different approaches, techniques, and mechanisms for overcoming various perspectives of smart cities' security. Krichen et al. [13] followed an approach of the model based on consisting of modelling the system with suitable formalism, derivation of suites from the model, applied criteria of convergence for selection of appropriate tests, execution of the tests, and lastly collection of verdicts and their analysis for detection of errors and debug them. The formalism adopted was based on the model of extended timed automata with inputs and outputs. Kaur and Saini [14] described the IoT security challenges, issues, and mechanisms. Le-Dang and Le-Ngoc [15] presented a detailed survey of the architectural design and key technologies of wireless communication for enabling applications of smart city. The study also elaborated the probable threats of security to the devices of IoT in the environment of smart city. Jaafar et al. [16] discussed the literature on the IoT-based smart city from the architecture, platform, technology, and application domain perspectives. Various challenges are raised to the best-effort IoT during the security, end-to-end communication, and energy efficiency. Szymanski [17] surveyed the weaknesses of security of best-effort IoT and, furthermore, presented a secure deterministic industrial tactile IoT core network.

Liu et al. [18] have addressed the attack of traffic analysis for smart homes where the opponents interrupt the traffic from and to the gateway of smart home and profile residents' behavior by digital traces. The traditional tools of cryptography are generally not feasible due to the usefulness of opponents. The study offered a framework for privacy preserved obfuscation for achieving this objective. Several simulations were performed for the effectiveness of the proposed framework. The results elaborated the effectiveness of the framework compared to the existing approaches. Alromaihi et al. [19] analyzed the privacy and security of the healthcare applications for smart cities. Firstly, the study provides a detailed review of the various applications of IoT and their cyber vulnerabilities and then presents a detailed assessment of potential approaches for mitigating the issues of cyber-attacks. Currently, various use cases are available for smart cities. These use cases are in the form of cooperative transportation network, autonomous vehicles, and smart roads for enhancing data propagation. Brincat et al. [20] presented an overview of the scenarios of IoT technology-based smart cities for the intelligent transportation

system. The study has presented the integration of cloud computing with the IoT for big data. The research is attempted for establishing security of the network architecture for enhancing the issues of security [21]. IoT devices are producing traffic based on specific features and variations with respect to traditional devices. Study has been presented for analyzing the possibility of applications of these features for classification of devices. Such classification is better in situation of heterogeneity and dynamic. Total of 41 devices of IoT were used [22]. Research was presented as a secure scenario for operating wireless mobile 6G network to manage big data in smart building [23].

Barbosa et al. [24] proposed a smart card cluster for ensuring message authentication and integration by using hardware signing. The approach is portable, modular, cost-effective, and flexible. With the support of results of the study, it is revealed that the approach outperformed existing solutions. Qureshi et al. [25] presented a framework of detecting version number attack, HELLO-Flood attack, Black hole attack, and Sinkhole attack. Various parameters were used for measuring performance of the framework. These parameters include the true positive rate, detection accuracy rate, false-positive rate, end-to-end delay, and throughput. The results revealed the support of the proposed framework for consideration in the environment of industrial Internet of things. Teng et al. [26] proposed a model of low-cost code dissemination which propagates the update code through mobile vehicles in the city with adoptable style of communication. For code stations, a coverage-based greedy deployment approach was used and algorithm of optimized code selection was used for maximizing code dissemination coverage over the city with low time and cost. Several experiments were performed for validation of the proposed study and the results revealed the effectiveness of the proposed approach. Accurate anomaly detection and identification based on the IoT for traffic identification can be considered as the essential issue for research to tackle. Shafiq et al. [27] proposed an approach of the bijective soft set for feature selection for selecting effective features and then devised approach of CorrACC feature selection metric. Furthermore, the research developed a new technique of features' selection algorithm Corracc based on CorrACC for extracting the most suitable and effective features for the classifier of machine learning through metric of ACC.

### 3. Analysing the Existing State-of-the-Art Work for Securing Data Traffic of Smart Devices and IoT

The security and privacy aspects are an insinuation of smart city. Research studies in the area have exploited the potential applications and their implications on smart devices. A small number of threats have gained more press in recent times than various ransomware campaigns. The malware and ransomware that encrypt files for denying data access till ransom can be paid by the owner [28]. Alternatives such as cryptolocker, reveton, wannacry, and cryptowall [29] are among the huge number of users paying ransoms. Including

a classic ransomware victim, such as loses of personal files or significant work extending from financial information to family pictures [30], there are various victim classes for whom harms are difficult to analyze. The study has presented a detailed survey which focuses on the IoT architecture security and facilitates a comprehensive taxonomy of key issues related with the area and main technologies. Appropriate protocols for infrastructure of IoT and open source tools and platforms for its development are discussed. Issues, challenges, and future directions are given [31]. Research work has been presented with the novel protocol in which the client of IoT can share part of the validation function with the server. Conflict of data at earlier is detected by such clients [32].

Various approaches have been practiced for solving various issues of smart cities. The purpose of this study was to identify the existing research studies in the area. Various popular libraries were searched for achieving the associated details of research. Figure 1 graphically represents the types of articles with the papers published in the ScienceDirect.

Figure 2 depicts the publication titles in the given library.

Figure 3 represents the year of publications. The figure reveals that more articles were published in the year 2020 which further shows that there is a significant growth in research in the given area.

Figure 4 depicts disciplines of the publications in the given library. It was shown in the figure that more articles were published in the field of Computer Science.

The Springer library was searched for analyzing the existing research in the area. Figure 5 briefly describes the disciplines covered in the area with total of publications.

Figure 6 describes the publication types with the total of publications in the given area.

The ACM library was searched for obtaining the relevant materials for the analysis process. Figure 7 depicts the publication types and papers in the given library.

Figure 8 depicts the conference held with the total of papers in the given area.

Figure 9 represents the types of contents in the given library.

The analysis was explored for further in-depth study of the materials in the given area of research. Figure 10 represents the media format with the publications in the library mentioned.

The IEEE library was searched and the associated results for analysis were obtained. Figure 11 depicts the publication types with the number of papers in the IEEE.

This library was further explored in order to obtain more results of the search. Figure 12 represents the location of conferences held.

The study analyzed the publication topics covered by the current study and identified a list of topics which are shown in Figure 13.

## 4. Results and Discussion

The experimental and simulation results are carried out on the dataset downloaded from Kaggle "Internet Firewall." The experimental work was performed using the Python tool.

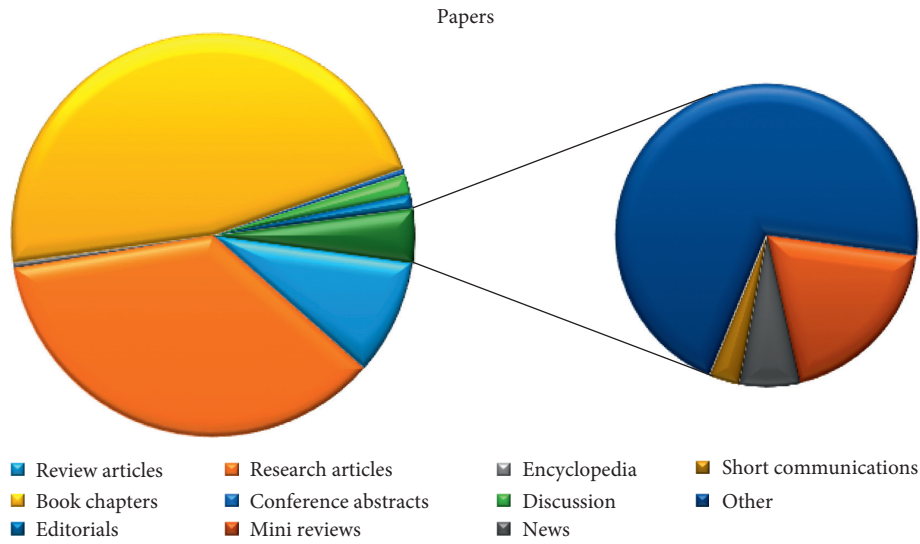


FIGURE 1: Article types.

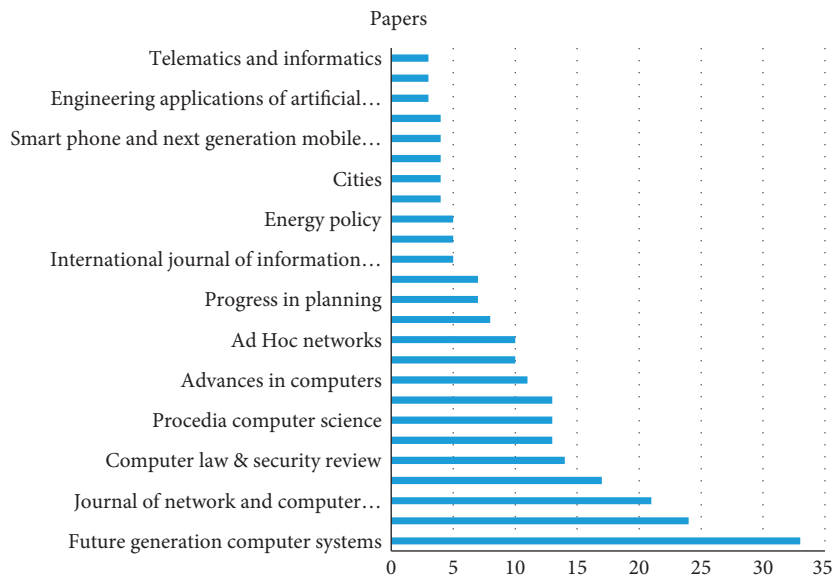


FIGURE 2: Publication titles.

This research work proposes the uses of a hybrid deep learning model based on convolution neural network (CNN) and support vector machine (SVM). The CNN is used for the classification purposes, while the SVM is used for the recognition and prediction purposes. The performance comparison in between these algorithms is depicted in Figure 14.

After assessing the planned hybrid model for various performance metrics such as misclassification rate, specificity, F measure, precision, recall, and accuracy, it was concluded that the hybrid model outperforms very well among other classification algorithms as depicted in Figure 15. The other two generic techniques random forest and decision rules are used to

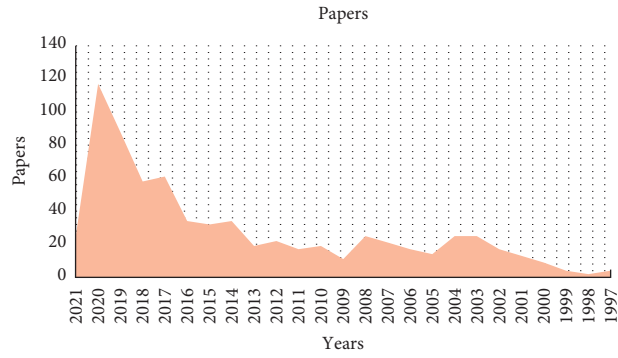


FIGURE 3: Years of publication.

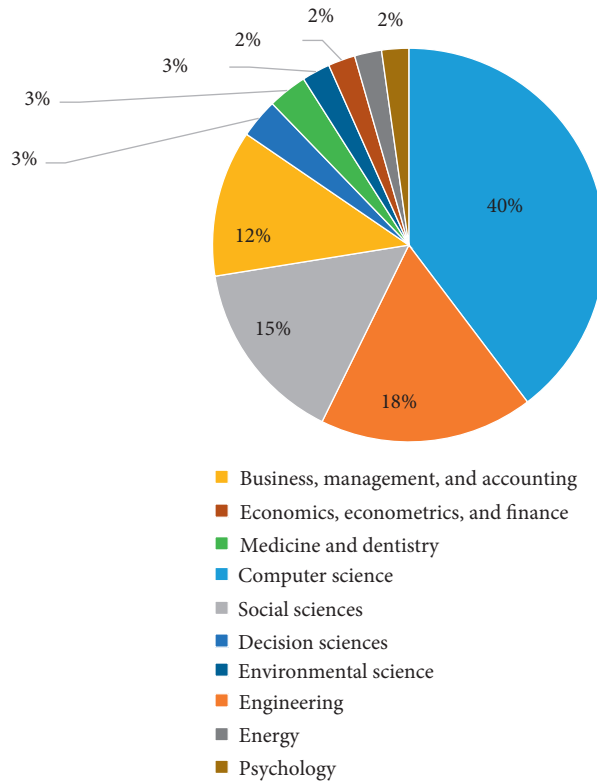


FIGURE 4: Disciplines of publication.

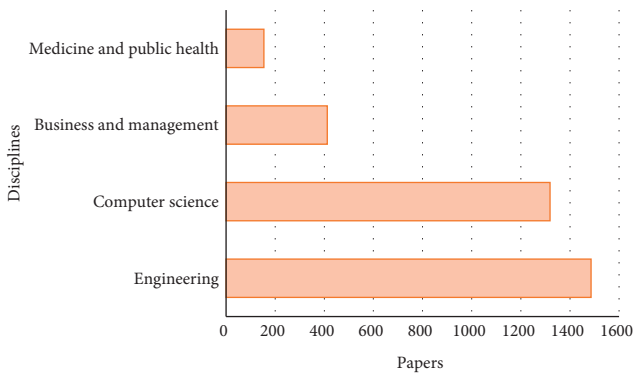


FIGURE 5: Disciplines and number of articles.

validate the applicability of the hybrid model in the recognition and classification task:

- (i) Hybrid model results: Figure 15 shows the experimental results of the proposed hybrid model. The CNN is considered as one of the best technique among deep architectures to accurately classify different objects.
- (ii) Random forest-based results: Figure 16 depicts the recognition capabilities of the random forest-based model based on different performance metrics.
- (iii) Decision rules: using different performance metrics, the recognition capabilities of decision rule-based recognition model is depicted in Figure 17.

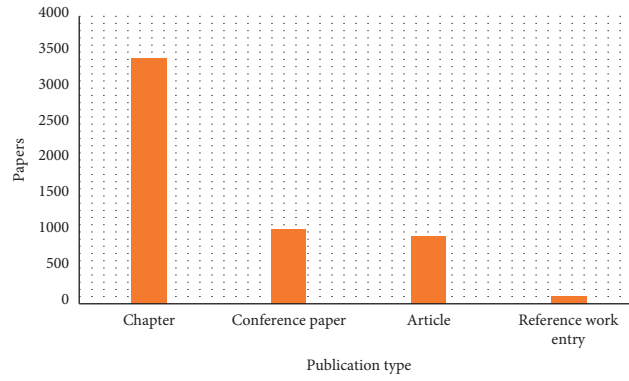


FIGURE 6: Publication types and papers.

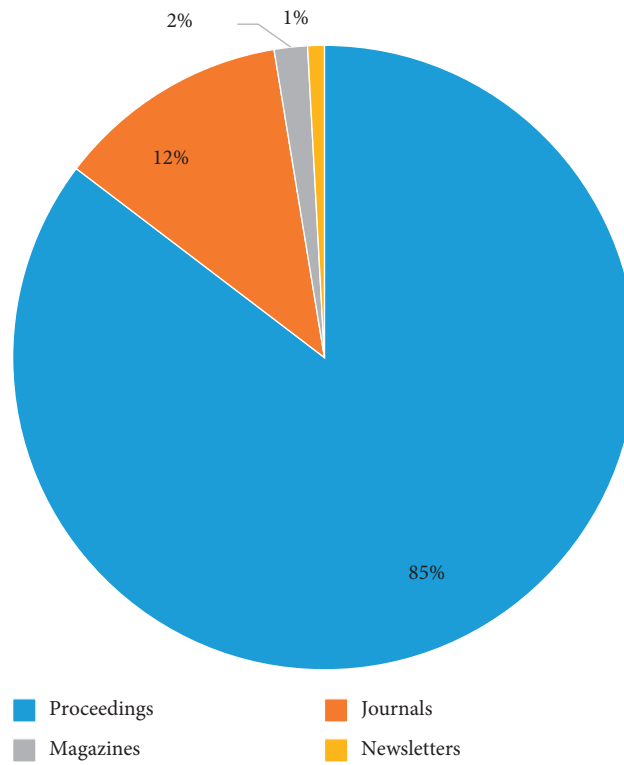


FIGURE 7: Publication types and papers.

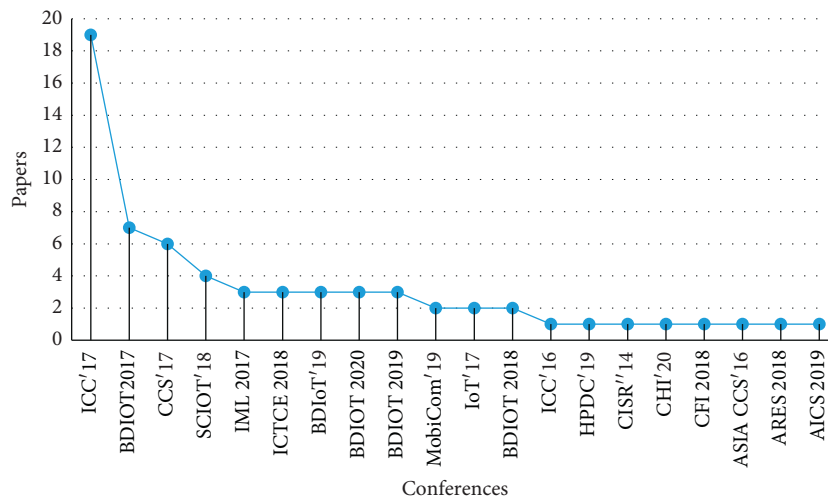


FIGURE 8: Conference location and papers.

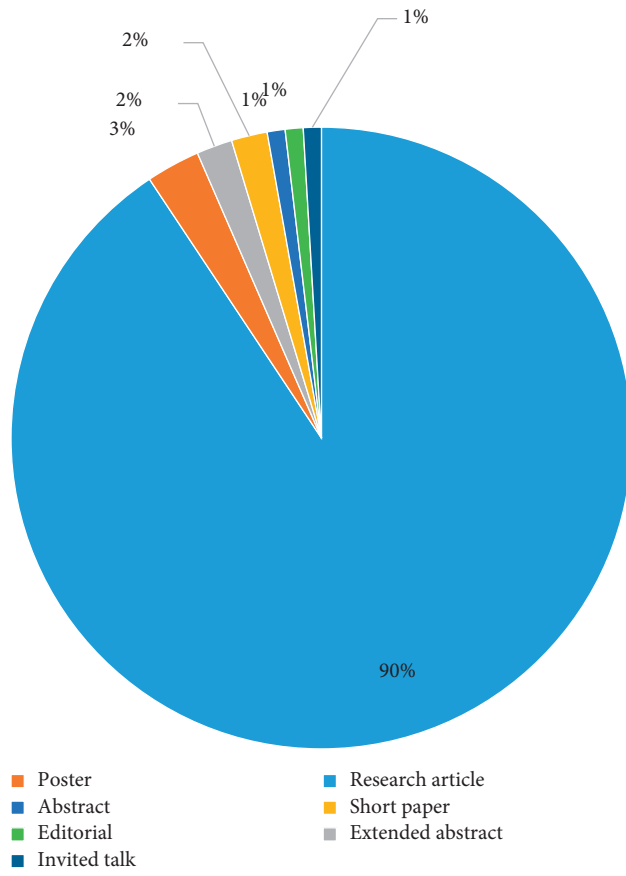


FIGURE 9: Content type.

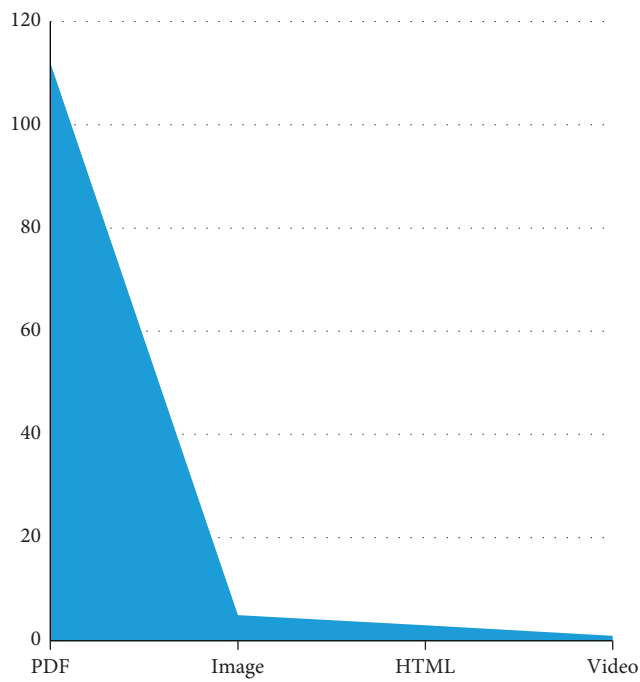


FIGURE 10: Media format.



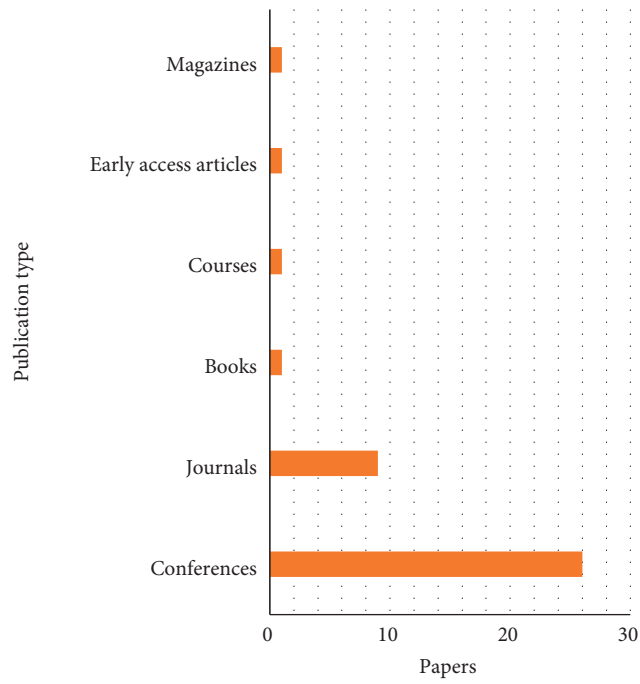


FIGURE 11: Publication type.



FIGURE 12: Conference locations.

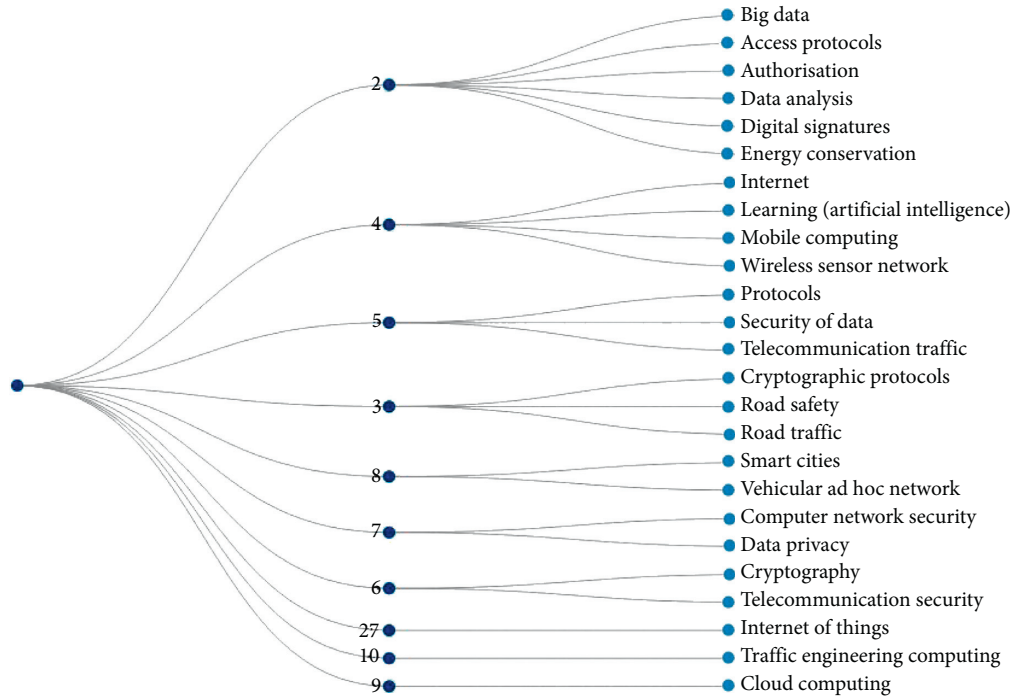


FIGURE 13: Publication topics and papers.

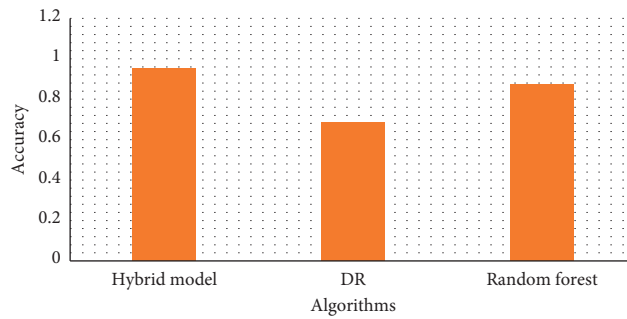


FIGURE 14: Comparison of algorithms for the planned study.

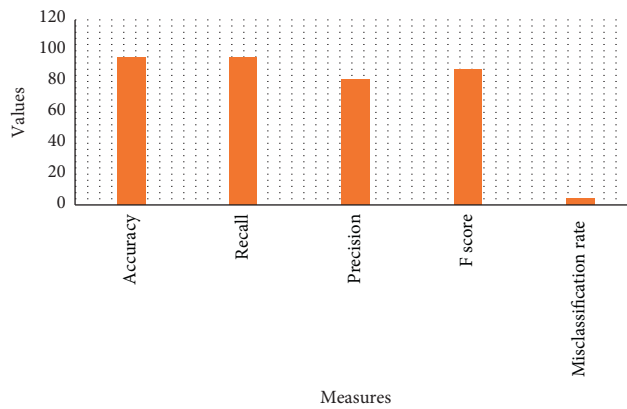


FIGURE 15: Hybrid model-based classification and recognition results.

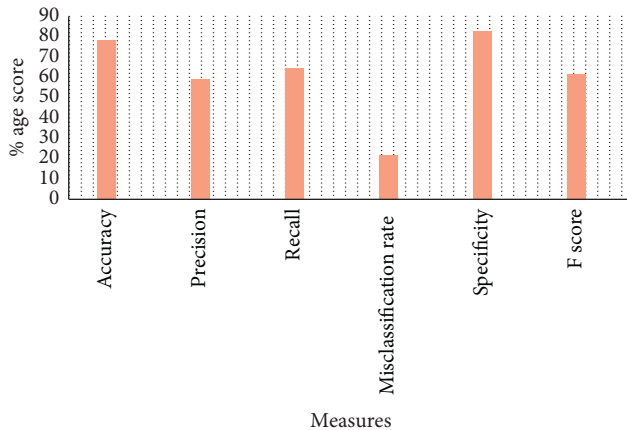


FIGURE 16: Random forest-based recognition results.

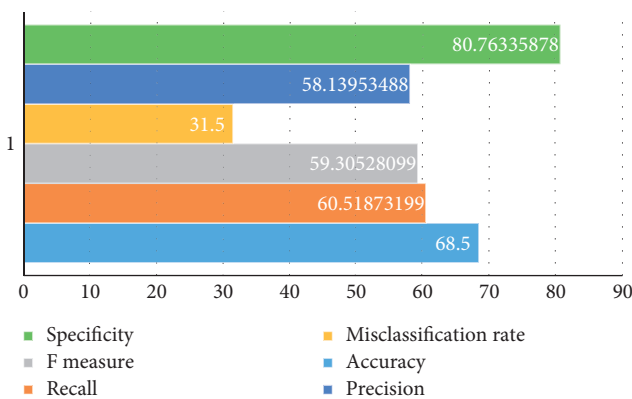


FIGURE 17: Decision rule-based recognition results.

## 5. Conclusion

Numerous devices such as sensor, actuator, and many other smart devices are connected and installed for communication, interaction, and solving the issues smart devices. The increase in the number of devices connected to the IoT is rising day by day. The connection of these devices has provided openings' directions for the smart applications which is one of the growing areas of research. Among these opportunities, security and privacy are considered to be one of the major issues for researchers to tackle. Smart communication is the direct need of modern societies. The role of IoT is understandable in the smart communication of these devices. The technology has mainly focused on efficient well-being of humans and with the protection of environment. These devices are communicating with each other and facilitating human life. The connection of these devices has provided openings' directions for the smart applications which is one of the growing areas of research. Proper security measures can prevent attackers from interrupting the security of IoT network inside the smart city for secure data traffic. Keeping in view the security consideration of data traffic for smart cities and IoT, the proposed study presented machine learning algorithms for securing the data traffic based on a firewall for smart devices and IoT network. The study has used the dataset of "Firewall" for validation

purposes. The experimental results of the approach shows that hybrid deep learning model (based on convolution neural network and support vector machine) outperforms than decision1 rules and random forest by generating a recognition rate of 95.5% for the hybrid model, 68.5% for decision rules, and 78.3% accuracy for random forest. The validity of the proposed model is also tested based on other performance metrics such as error rate, recall, f score, and precision. This great accuracy rate and other performance values show the influence of the proposed hybrid model for secure data traffic purposes in smart devices.

## Data Availability

The data used in this study are not available.

## Conflicts of Interest

The authors declare no conflicts of interest.

## References

- [1] Z. Gu, S. Nazir, C. Hong, and S. Khan, "Convolution neural network based higher accurate intrusion identification system for the network security and communication," *Security and Communication Networks*, vol. 2020, Article ID 8830903, 10 pages, 2020.
- [2] H. U. Rahman, A. U. Rehman, S. Nazir, I. U. Rehman, and N. Uddin, "Privacy and security—limits of personal information to minimize loss of privacy," in *Proceedings of the Presented at the Future of Information and Communication Conference*, San Francisco, CA, USA[Online]. Available: [https://link.springer.com/chapter/10.1007/978-3-030-12385-7\\_65#citeas](https://link.springer.com/chapter/10.1007/978-3-030-12385-7_65#citeas), San Francisco, CA, USA, March 2019.
- [3] J. Zhang, S. Nazir, A. Huang, and A. Alharbi, "Multicriteria decision and machine learning algorithms for component security evaluation: library-based overview," *Security and Communication Networks*, vol. 2020, Article ID 8886877, 14 pages, 2020.
- [4] B. Liao, Y. Ali, S. Nazir, L. He, and H. U. Khan, "Security analysis of IoT devices by using mobile computing: a systematic literature review," *IEEE Access*, vol. 8, pp. 120331–120350, 2020.
- [5] L. Ning, Y. Ali, H. Ke, S. Nazir, and Z. Huanli, "A hybrid MCDM approach of selecting lightweight cryptographic cipher based on ISO and NIST lightweight cryptography security requirements for Internet of Health Things," *IEEE Access*, vol. 8, pp. 220165–220187, 2020.
- [6] L. Wang, Y. Ali, S. Nazir, and M. Niazi, "ISA evaluation framework for security of internet of health things system using AHP-TOPSIS methods," *IEEE Access*, vol. 8, pp. 152316–152332, 2020.
- [7] S. Sivagurunathan, A. Sebastian, and K. Prathapchandran, "Internet of things for developing smart sustainable cities (SSC): a security perspective," in *Connectivity Frameworks for Smart Devices*, pp. 307–331, Springer, New York, NY, USA, 2016.
- [8] J. M. Ibrahim, A. Karami, and F. Jafari, "A secure smart home using internet-of-things," in *Proceedings of the 9th International Conference on Information Management and Engineering*, pp. 69–74, Barcelona, Spain, October 2017.
- [9] M. Wittl and D. Konstantas, "A secure and privacy-preserving Internet of Things framework for smart city," in *Proceedings of*

- the 6th International Conference on Information Technology: IoT and Smart City*, pp. 145–150, Hong Kong, December 2018.
- [10] X. Jia, X. Li, and Y. Gao, “A novel semi-automatic vulnerability detection system for smart home,” in *Proceedings of the International Conference on Big Data and Internet of Thing*, pp. 195–199, London, UK, December 2017.
  - [11] M. Talal, A. A. Zaidan, B. B. Zaidan et al., “Smart home-based IoT for real-time and secure remote health monitoring of triage and priority system using body sensors: multi-driven systematic review,” *Journal of Medical Systems*, vol. 43, no. 3, p. 42, 2019.
  - [12] P. K. Sharma, J. H. Park, Y.-S. Jeong, J. H. Park, and Applications, “Shsec: sdn based secure smart home network architecture for internet of things,” *Mobile Networks and Applications*, vol. 24, no. 3, pp. 913–924, 2019.
  - [13] M. Krichen, M. Lahami, O. Cheikhrouhou, R. Alroobaea, and A. J. Maâlej, “Security testing of internet of things for smart city applications: a formal approach,” in *Smart Infrastructure and Applications*, pp. 629–653, Springer, New York, NY, USA, 2020.
  - [14] G. Kaur and K. S. Saini, “Securing network communication between motes using hierarchical group key management scheme using threshold cryptography in smart home using internet of things,” in *Computing and Network Sustainability*, pp. 201–212, Springer, New York, NY, USA, 2017, Lecture Notes in Networks and Systems.
  - [15] Q. Le-Dang and T. Le-Ngoc, “Internet of things (IoT) infrastructures for smart cities,” in *Handbook of Smart Cities*, pp. 1–30, Springer, New York, NY, USA, 2018.
  - [16] A. A. Jaafar, K. H. Sharif, M. I. Ghareb, and D. N. A. Jawawi, “Internet of thing and smart city: state of the art and future trends,” in *Advances in Computer Communication and Computational Sciences*, pp. 3–28, Springer, New York, NY, USA, 2019.
  - [17] T. H. Szymanski, “Securing the industrial-tactile internet of things with deterministic silicon photonics switches,” *IEEE Access*, vol. 4, pp. 8236–8249, 2016.
  - [18] J. Liu, C. Zhang, and Y. Fang, “EPIC: a differential privacy framework to defend smart homes against internet traffic analysis,” *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 1206–1217, 2018.
  - [19] S. Alromaihi, W. Elmedany, and C. Balakrishna, “Cyber security challenges of deploying IoT in smart cities for healthcare applications,” in *Proceedings of the 2018 6th International Conference on Future Internet of Things and Cloud Workshops (FiCloudW)*, pp. 140–145, Barcelona, Spain, August 2018.
  - [20] A. A. Brincat, F. Pacifici, S. Martinaglia, and F. Mazzola, “The internet of things for intelligent transportation systems in real smart cities scenarios,” in *Proceedings of the 2019 IEEE 5th World Forum on Internet of Things (WF-IoT)*, pp. 128–132, Limerick, Ireland, April 2019.
  - [21] C. Stergiou, K. E. Psannis, B. B. Gupta, Y. Ishibashi, and Systems, “Security, privacy & efficiency of sustainable cloud computing for big data & IoT,” *Sustainable Computing: Informatics and Systems*, vol. 19, pp. 174–184, 2018.
  - [22] I. Cvitić, D. Peraković, and M. Periša, “Ensemble machine learning approach for classification of IoT devices in smart home,” in *Proceedings of the 3rd International Conference on Data Intelligence and Security*, pp. 1–24, Texas, TX, USA, June 2020.
  - [23] C. L. Stergiou and K. E. Psannis, *IoT-based Big Data Secure Management in the Fog over a 6G Wireless Network*, <https://ieeexplore.ieee.org/document/9239366>, 2020.
  - [24] G. Barbosa, P. T. Endo, and D. Sadok, “An internet of things security system based on grouping of smart cards managed by field programmable gate array,” *Computers & Electrical Engineering*, vol. 74, pp. 331–348, 2019.
  - [25] K. N. Qureshi, S. S. Rana, A. Ahmed, and G. Jeon, “A novel and secure attacks detection framework for smart cities industrial internet of things,” *Sustainable Cities and Society*, vol. 61, Article ID 102343, 2020.
  - [26] H. Teng, Y. Liu, A. Liu et al., “A novel code data dissemination scheme for Internet of Things through mobile vehicle of smart cities,” *Future Generation Computer Systems*, vol. 94, pp. 351–367, 2019.
  - [27] M. Shafiq, Z. Tian, A. K. Bashir, X. Du, and M. Guizani, “IoT malicious traffic identification using wrapper-based feature selection mechanisms,” *Computers & Security*, vol. 94, Article ID 101863, 2020.
  - [28] B. A. S. Al-rimy, M. A. Maarof, and S. Z. M. Shaid, “Ransomware threat success factors, taxonomy, and countermeasures: a survey and research directions,” *Computers and Security*, vol. 74, pp. 144–166, 2018.
  - [29] N. Hampton and Z. A. Baig, *Ransomware: Emergence of the Cyber-Extortion Menace*, Cowan University Joondalup Campus, Perth, Australia, 2015.
  - [30] G. O’Gorman and G. McDonald, *Ransomware: A Growing Menace*, Symantec Corporation, Arizona, AZ, USA, 2012.
  - [31] B. Gupta and M. J. C. Quamara, “C. Practice, and experience an overview of internet of things (IoT): architectural aspects, challenges, and protocols,” vol. 32, no. 21, Article ID e4946, 2020.
  - [32] A. Al-Qerem, M. Alauthman, A. Almomani, and B. B. Gupta, “IoT transaction processing through cooperative concurrency control on fog-cloud computing environment,” *Soft Computing*, vol. 24, no. 8, pp. 5695–5711, 2020.