DeFFusion: CNN-based Continuous Authentication Using Deep Feature Fusion

YANTAO LI, PENG TAO, and SHAOJIANG DENG, Chongqing University, China
GANG ZHOU, Department of Computer Science, William & Mary, USA

Smartphones have become crucial and important in our daily life, but the security and privacy issues have been the major concerns of smartphone users. In this paper, we present DeFFusion, a CNN-based continuous authentication system using Deep Feature Fusion for smartphone users, by leveraging the accelerometer and gyroscope ubiquitously built into smartphones. With the collected data, DeFFusion first converts the time domain data into frequency domain data using the fast Fourier transform and then inputs both of them into a designed CNN, respectively. With the CNN-extracted features, DeFFusion conducts the feature selection utilizing factor analysis and exploits balanced feature concatenation to fuse these deep features. Based on the one-class SVM classifier, DeFFusion authenticates the current users as a legitimate user or an impostor. We evaluate the authentication performance of DeFFusion in terms of impact of training data size and time window size, accuracy comparison on different features over different classifiers and on different classifiers with the same CNN-extracted features, accuracy on unseen users, time efficiency, and comparison with representative authentication methods. The experimental results demonstrate that DeFFusion performs the best accuracy, by achieving the mean equal error rate of 1.00% in 5-second time window size.

CCS Concepts: • **Security and privacy** → **Biometrics**; • **Human-centered computing** → *Collaborative and social computing devices*; • **Computing methodologies** → *Neural networks.*

Additional Key Words and Phrases: Continuous authentication, deep feature fusion, CNN, factor analysis, OC-SVM, EER

ACM Reference Format:
Yantao Li, Peng Tao, Shaojiang Deng, and Gang Zhou. 2021. DeFFusion: CNN-based Continuous Authentication Using Deep Feature Fusion. *ACM Trans. Sensor Netw.* 1, 1, Article 1 (September 2021), 21 pages. https://doi.org/10.1145/3397179

1 INTRODUCTION

Driven by new mobile and communication technologies, such as the mobile Internet, the Internet of Things, artificial intelligence, and economic and social developments, mobile smart devices have been developing rapidly. They have transformed from communication-only devices to diversified social entertainment tools, and have been bringing great convenience to people's daily life, work, and commercial activities [1, 2]. For example, people can perform daily leisure and work activities, such as e-book reading, online video watching, and electronic document processing through smart tablets, and conduct entertainment and business activities, such as online shopping, mobile payments, and e-banking services (payment and transfer) through smartphones. As mobile devices

Authors' addresses: Yantao Li; Peng Tao; Shaojiang Deng, Chongqing University, 174 Shapingba Central St, Chongqing, China, 400044, yantaoli@cqu.edu.cn,sj_deng@cqu.edu.cn; Gang Zhou, Department of Computer Science, William & Mary, 251 Jamestown Rd, Williamsburg, VA, 23185, USA, gzhou@cs.wm.edu.

1550-4859/2021/9-ART1 $15.00
https://doi.org/10.1145/3397179

continue to infiltrate all aspects of people's daily life and work, more and more private information and confidential data are stored on smartphones by users, such as personal emails and photos, social network accounts, electronic bank accounts, and even confidential trade information. Therefore, there is an increasing need for security mechanisms to authenticate the smart device users for protecting users' personal and privacy data on smart devices [3, 4]. The existing smart devices are preliminary equipped with one-time security mechanisms, which are typically the only defense available on smart devices and can be broadly categorized into traditional security mechanisms and physiological biometrics-based security mechanisms [5]. However, traditional security mechanisms, such as easy-to-remember PINs and quick-to-draw graphical patterns, are susceptible to guessing [6] and eye glimpsing [7]. Physiological biometrics-based security mechanisms, such as fingerprints (touch ID) [8] and face recognition (face ID) [9], are prone to smudge [10] and video capture attacks [11]. Moreover, one-time security mechanisms only perform at the time of user initial logging-in, which raises security concerns that attackers can easily get access to everything on smart devices bypassing the unlocking [12].

To address these security deficiencies, the continuous authentication mechanisms have been investigated by researchers. The continuous authentication mechanisms are able to frequently authenticate smart device users via behavioral biometrics-based approaches. These approaches utilize user behavioral patterns, such as touch gestures [13], gait [14], and GPS patterns [15]. To extract more robust and distinctive behavioral features, comparing with traditional approaches, the deep learning models especially the convolutional neural networks (CNNs) have illustrated broad superiority [16, 17], on behavioral biometrics, such as voice [18], signature [19, 20], gait [14, 21], and keystroke [22]. They can automatically learn high-level representative features from input data and extract discriminative features as the outputs [23–25]. In particular, the authors use two deep CNN architectures for iris recognition, one with eight convolutional layers and the other with five convolutional and two inception layers [26]. However, most of these approaches consisting of many layers and channels require much computational budget. Thus, it is challenging to design a light-weight and effective CNN for discriminative feature extraction. In addition, feature fusion strategy can combine dissimilar sensor features to generate a fused feature vector, which greatly improves the authentication accuracy [27–29]. Biometric feature fusion can generally be divided into sensor-level fusion [30], feature-level fusion [27, 29], score-level fusion [31], and decision-level fusion [32]. Especially, the authors use serial and parallel feature fusion strategies to combine two different types of features for high authentication accuracy [29]. Nevertheless, most of these feature fusion strategies directly combine two or more modalities of features for the authentication accuracy improvement. Hence, it is also challenging to explore an efficient feature fusion strategy for multi-model deep features.

To overcome these challenges, we specially design a light-weight CNN based on the basic block (a bottlenet unit with depthwise convolution) and down block (a basic block for spatial down sampling) to learn and extract discriminative deep features, and exploit a balanced feature concatenation to fuse the selected deep features in a continuous authentication system. In this paper, we present DeFFusion, a CNN-based continuous authentication system using Deep Feature Fusion, which leverages the accelerometer and gyroscope on smartphones to capture users' behavioral patterns. Specifically, DeFFusion consists of five modules: data collection, data preprocessing, deep feature extraction, classification, and authentication. The operation of DeFFusion is composed of two phases of the enrollment phase and the continuous authentication phase. In the enrollment phase, DeFFusion exploits the collected sensor data from the accelerometer and gyroscope to train the designed CNN, extract CNN-based features, then fuse the deep features by the balanced feature concatenation, and finally train the one-class SVM classifier, thereby learning a profile of the legitimate user. In the continuous authentication phase, with the trained CNN and classifier, DeFFusion

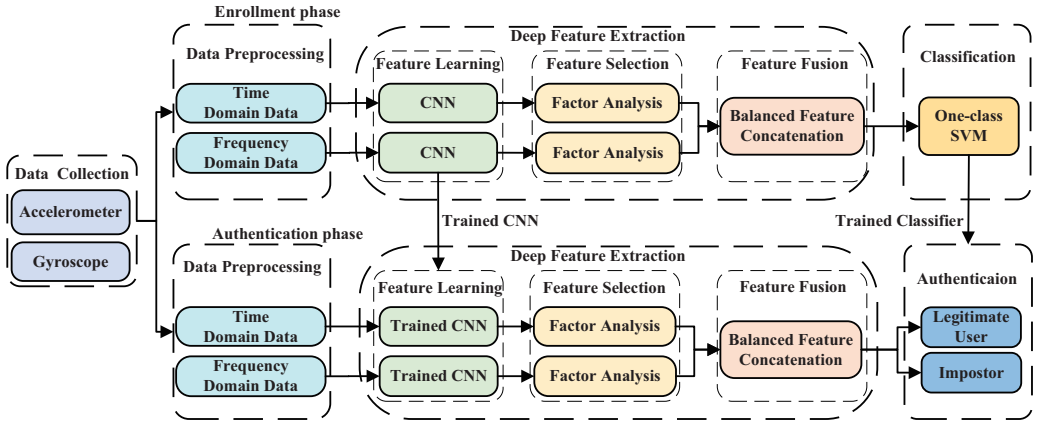

Fig. 1. Architecture of DeFFusion

classifies the current user as the legitimate user or an impostor based on the accelerometer and gyroscope data, thereby ensuring only the owner can access the smartphone. We consider a scenario that an attacker tries to access an unattended smartphone right after the owner logs in the system, but has no knowledge of the initial login information.

The main contributions of this work are summarized as follows.

- We propose DeFFusion, a CNN-based continuous authentication system using deep feature fusion for smartphone users, by leveraging the accelerometer and gyroscope on smartphones. DeFFusion consists of five modules: data collection, data preprocessing, deep feature extraction, classification, and authentication.
- We specially design a CNN based on the basic block and down block to learn and extract discriminative deep features for the time and frequency domain data. We utilize the balanced feature concatenation to fuse the CNN-extracted features selected by the factor analysis.
- We evaluate the authentication performance of DeFFusion and the experimental results indicate that DeFFusion reaches the mean EER of 1.00% in 5-second time window size, showing the best accuracy comparing with representative schemes.

The rest of this work is organized as follows: Section 2 presents the overview of DeFFusion. In Section 3, we introduce data collection and data preprocessing for deep feature extraction. Section 4 details the CNN-based feature extraction approach consisting of the deep feature learning, selection and fusion. We elaborate the authentication with one-class SVM classifier in Section 5. In Section 6, we describe the details of experimental settings and extensively evaluate the performance of DeFFusion. We review the recognition systems on deep learning and biometric fusion in Section 7 and conclude this work in Section 8.

2 DEFFUSION OVERVIEW

In this section, we present the overview of the CNN-based continuous authentication system using deep feature fusion, DeFFusion. From the architecture of DeFFusion illustrated in Fig. 1, DeFFusion is composed of two phases: the enrollment phase and the continuous authentication phase. Specifically, in the enrollment phase, DeFFusion learns the profile of a legitimate user by utilizing the training data to train the CNN and the classifier, and then authenticates users by exploiting the trained CNN and trained classifier on the testing data in the continuous authentication phase.

DeFFusion is composed of five modules: data collection, data preprocessing, deep feature extraction, classification, and authentication. More specifically, the data collection module employs the smartphone built-in sensors of the accelerometer and gyroscope to capture users' every subtle operation behaviors on their phones and sample the corresponding behavioral data on smartphones instantaneously. The data preprocessing module converts the collected time domain data into the frequency domain data by using the fast Fourier transform and then applies StandardScaler on the time and frequency domain data. The deep feature extraction module is composed of the deep feature learning, feature selection, and feature fusion. Moreover, the deep feature extraction module learns discriminative features based on the designed convolutional neural network (CNN), then selects representative features by factor analysis for the preprocessing time domain data and frequency domain data, respectively, and finally fuses the corresponding deep features through a balanced feature concatenation. With the fused deep features, the classification module trains the one-class SVM (OC-SVM) classifier to generate the legitimate user's profile from the training data. Based on the trained OC-SVM and testing data, the authentication module classifies the current user as a legitimate user or an imposter. Finally, DeFFusion will allow the continuous usage of the smartphone if it is a legitimate user and meanwhile continuously authenticate the user; otherwise, it requires the initial login inputs. Our work is different from the others in that: 1) specially design a light-weight CNN to learn and extract discriminative deep features, and 2) exploit a balanced feature concatenation to fuse the selected deep features.

3 DATA COLLECTION AND PREPROCESSING

In this section, we present the data collection and data preprocessing modules of DeFFusion, respectively.

3.1 Data Collection

We select two motion sensors - the accelerometer and gyroscope built-in smartphones to capture a user's coarse-grained motion patterns, such as arm movements and gaits, and fine-grained motion patterns, such as touch gestures on screens, respectively. Note that we consider a general scenario where users' operation on smartphones can be captured by the two sensors in this work [33].

Once a user operates the smartphone, the data collection module starts to collect the raw sensor data from the accelerometer and gyroscope, respectively, for a time period t with a sampling rate f_s. For a time period t, n ($n = t \times f_s$) samples of raw accelerometer and gyroscope sensor data can be collected. Each synchronized sample can be denoted as $(x_a, y_a, z_a, x_g, y_g, z_g)^T \in \mathbb{R}^6$, where x, y, z indicate the three axes of a sensor, and a, g represent the accelerometer and gyroscope, respectively.

3.2 Data Preprocessing

For a time period t, DeFFusion can collect n samples of time domain data for the accelerometer and gyroscope. The time domain data can be represented by a $d \times n$ matrix $\mathbf{T_t} = (\mathbf{x_a}, \mathbf{y_a}, \mathbf{z_a}, \mathbf{x_g}, \mathbf{y_g}, \mathbf{z_g})^T$, where $d = 6$, $n = t \times f_s$, and $\mathbf{x_a} = (x_{a,1}, x_{a,2}, ..., x_{a,n})$ for x-axis samples of the accelerometer, by using a row vector to denote one-axis samples of a sensor. For the frequency domain data, we apply the fast Fourier transform to $\mathbf{T_t}$ and then sequentially stack the real part and imaginary part of each element in $\mathbf{T_t}$ to construct a $d \times 2n$ matrix $\mathbf{F_t} = (\mathbf{x_{ar}}, \mathbf{x_{ai}}, \mathbf{y_{ar}}, \mathbf{y_{ai}}, \mathbf{z_{ar}}, \mathbf{z_{ai}}, \mathbf{x_{gr}}, \mathbf{x_{gi}}, \mathbf{y_{gr}}, \mathbf{y_{gi}}, \mathbf{z_{gr}}, \mathbf{z_{gi}})^T$, where r and i indicate the real part and imaginary part, respectively.

In order to obtain more standard data, we apply StandardScaler on each row (e.g. $\mathbf{x_a}$) of the time domain data $\mathbf{T_t}$, and each row (e.g. $\mathbf{x_{ar}}$) of the frequency domain data $\mathbf{F_t}$.

The preprocessed time domain data $\mathbf{T_t}$ and frequency domain data $\mathbf{F_t}$ are obtained and then used as the inputs of a convolutional neural network, respectively.

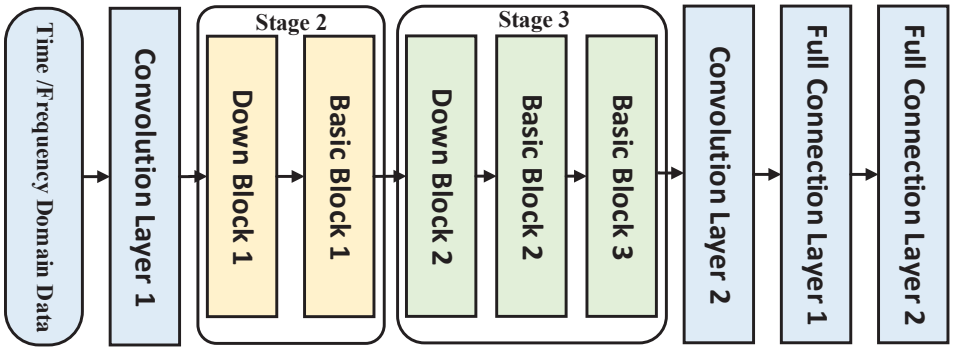

Fig. 2. Architecture of CNN

4 DEEP FEATURE EXTRACTION

In this section, we present a CNN-based deep feature extraction approach that consists of the deep feature learning, deep feature selection, and deep feature fusion.

4.1 Deep Feature Learning

We first elaborate the architecture of the designed CNN, and then detail the CNN-based feature learning for the preprocessed time domain data and frequency domain data, respectively.

4.1.1 CNN Architecture. We design an architecture of a convolutional neural network inspired by ShuffleNet [34, 35] as illustrated in Fig. 2 and Table 1. With the designed CNN architecture, the features in the time domain involving a number of temporal dynamics patterns and that in the frequency domain including spatial patterns in neighboring frequencies can be extracted, respectively. As shown in Fig. 2, the CNN architecture comprises the Convolution Layer 1, Stage 2 with the Down Block 1 and the Basic Block 1, Stage 3 with Down Block 2, Basic Block 2 and Basic Block 3, Convolution Layer 2, Full Connection Layer 1, and Full Connection Layer 2. Then, we detail the structures of the Basic Block and Down Block, respectively.

Basic Block structure is based on the bottlenet unit with depthwise convolution (DWConv) [36, 37], as illustrated in Fig. 3. Specifically, with the inputs, Basic Block starts with splitting the channels C into two identical branches. One branch remains as identity with $C/2$ channels. The other branch involves three convolutions with the same input and output channels, where the first 1×1 convolution (1×1 Conv) as the expansion layer increases the channel dimensions, the efficient 3×3 depthwise convolution (3×3 DWConv) applies a single filter for each input channel (input depth), and the last 1×1 convolution (1×1 Conv) combines the outputs of the DWConv to decrease the channel dimensions for matching that of the first branch. Batch normalization (BN) and rectified linear unit (ReLU) nonlinearity are applied to this branch except the DWConv (with BN only). Then, the outputs of the two branches are concatenated and then divided into g subgroups. Finally, the channel shuffle operation is applied by reshaping the output channel dimension into (g, n), transposing and flattening it back as the basic block outputs, where $C = g \times n$.

Down Block structure is based on a basic block for spatial down sampling [34, 35], as demonstrated in Fig. 4. Specifically, with the inputs, Down Block begins in two branches, each with $C/2$ channels. One branch is composed of the 1×6 DWConv with stride of 2 and BN, and 1×1 Conv with BN and ReLU. The other branch consists of 1×1 Conv, 1×6 DWConv with stride of 2, and 1×1 Conv. BN and ReLU nonlinearity are applied to this branch without ReLU on DWConv. Then, the two-branch

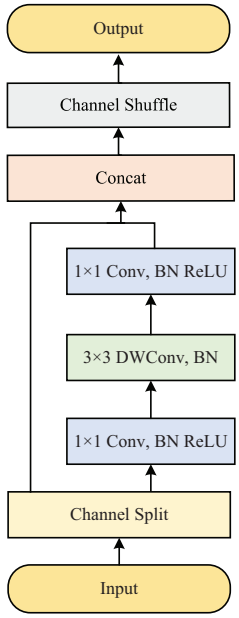

Fig. 3. Basic Block

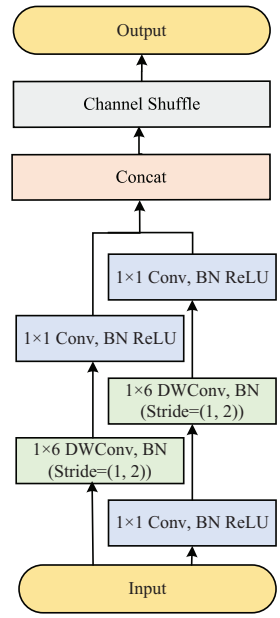

Fig. 4. Down Block

Table 1. CNN Body Architecture for Time (Frequency) Domain Data

Layer	Output	#Kernel	KSize	Stride	Padding
Sensor	$1 \times 20 \times 150(300)$	-	-	-	
Conv 1	$24 \times 20 \times 150(300)$	24	(3,3)	(1,1)	(1,1)
Stage 2	$32 \times 20 \times 73(148)$	32	(1,6)	(1,2)	(0,0)
	$32 \times 20 \times 73(148)$	32	(3,3)	(1,1)	(1,1)
Stage 3	$32 \times 20 \times 34(72)$	32	(1,6)	(1,2)	(0,0)
	$32 \times 20 \times 34(72)$	32	(3,3)	(1,1)	(1,1)
	$32 \times 20 \times 34(72)$	32	(3,3)	(1,1)	(1,1)
Conv 2	$64 \times 6 \times 34(72)$	64	(1,1)	(1,1)	(0,0)
AvgPool	$64 \times 6 \times 6(6)$	-	-	-	-
FC 1	$1 \times 3072(3072)$	-	-	-	-
FC 2	$1 \times 512(512)$	-	-	-	-

outputs are concatenated with C channels. Finally, we apply the channel shuffle operation on the concatenated outputs as the down block outputs.

4.1.2 CNN-based Feature Learning. Based on the designed CNN, we provide a deep feature extraction approach to learn discriminative features for the time domain data $\mathbf{T_t}$ and frequency domain data $\mathbf{F_t}$, respectively.

As demonstrated in Table 1, the input of the CNN can be the time domain data $\mathbf{T_t}$ or frequency domain data $\mathbf{F_t}$. The time domain data $\mathbf{T_t}$ in 5 seconds have 3000 ($5 \times 100 \times 6$) samples, which are reshaped as 20×150. In the first convolutional layer (Conv 1), there are 24 filters with the size of 3×3. In Stage 2, we apply the down block (Down Block 1) involving $C = 32$ filters with the size of

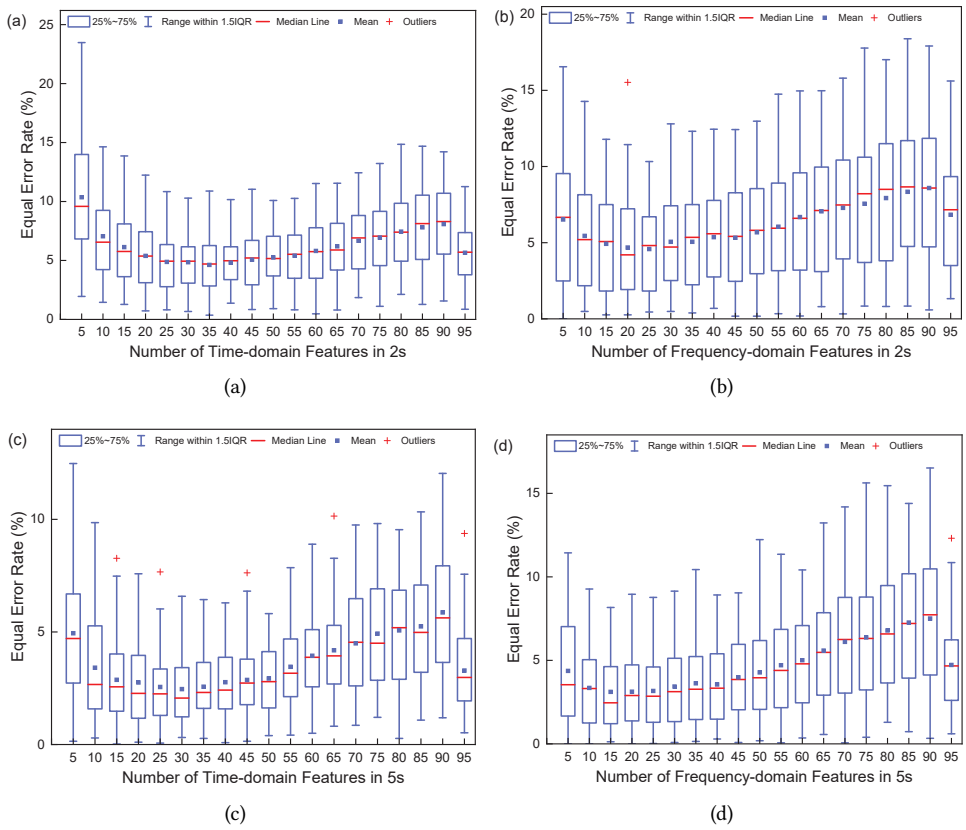

Fig. 5. EER for DeFFusion on different number of features in different time periods (2 seconds or 5 seconds)

1×6, and with $g = 2$, and then basic block (Basic Block 1) including $C = 32$ filters with the size of 3×3. In Stage 3, we apply the down block (Down Block 2) including 32 filters with the size of 1×6, and with $g = 2$, and then two basic blocks (Basic Block 2 and Basic Block 3) both containing 32 filters with the size of 3×3. In the fourth convolutional layer (Conv 2), there are 64 filters with the size of 1×1. The average pooling layer (AvgPool) with parameters $(1, 5)$ is exploited to decrease the channel output dimensions and extract features. Finally, we use two fully convolution layers (FC 1 and FC 2) to classify the inputs into a finite number of classes. For the frequency domain data $\mathbf{F_t}$ in 5 seconds, there are 6000 ($2 \times 5 \times 100 \times 6$) samples, which are reshaped as 20×300. Since the CNN architecture for the frequency domain is the same as that of the time domain, we list the different parameters in parentheses in Table 1. The AvgPool has the parameters of $(1, 12)$ for frequency domain. Note that the symbol "-" in the table indicates no corresponding parameter value. We set the CNN-learned feature number as 95 for both time and frequency domain.

4.2 Deep Feature Selection

Based on the CNN-learned deep features, we exploit the Factor Analysis (FA) to select discriminative features for the time domain data and frequency domain data, respectively.

We conduct experiments to investigate the optimal feature numbers for the time domain data and frequency domain data, respectively, thereby DeFFusion achieving the best accuracy. We compute

Table 2. Feature Number Reaching the Lowest EERs in Time and Frequency Domain over Different Time Periods

Time Period	Time domain	Frequency domain	# Feature
2s	35	20	55
5s	30	15	45

Table 3. Balanced Feature Number for DeFFusion in Different Time Periods

Time Period	Time domain	Frequency domain
2s	29	21
5s	28	22

the EERs of DeFFusion as feature numbers increase from 5 to 95 with a stride 5, for the time domain data and frequency domain data, respectively. Fig. 5 depicts the box plots of EERs for DeFFusion on different number of features over 2 and 5 seconds. As illustrated in Fig. 5, the EERs generally first decrease with the increase of the selected features until an optimal number, and then slightly increase until a sharply drop at 95. In addition, we list the feature numbers reaching the lowest EERs in the time and frequency domain over different time periods in Table 2. As listed in Table 2, with 2s-sampling data, 35 time-domain features and 20 frequency-domain features reach their lowest EERs, respectively, with the total feature number of 55. In 5-second time period, 30 features for time domain and 15 for frequency domain achieve their lowest EERs, respectively, with 45 features in total.

Considering the feature numbers reaching the lowest EERs and the time period, we select 50 deep features in total for the time domain and frequency domain data by using the factor analysis.

4.3 Deep Feature Fusion

To improve the performance of the DeFFusion authentication, we apply the balanced feature concatenation fusion after feature selection. The time domain feature can be represented by a vector $DeF_t[m]$, where m indicates the number of the deep features, and the frequency domain feature can be denoted by $DeF_f[n]$. Therefore, the balanced feature concatenation fusion $DeFF$ can be expressed as Eq. (1):

$$DeFF = [DeF_t[m], DeF_f[n]]. \tag{1}$$

Here, m and n balance the deep feature fusion thereby achieving the best accuracy.

Given $m + n = 50$ from the deep feature selection, we balance the m for the time domain features and n for the frequency domain features. We calculate the EERs of DeFFusion with different combinations (m, n) of time and frequency features over 2 and 5 seconds varying from (0,50) to (50,0) with stride (1,-1), respectively, as demonstrated in Fig. (6). As shown in Fig. (6), the EERs generally first decrease and then increase with the variation of the time and frequency combinations for 2 and 5 time periods, respectively. In addition, we tabulate the balanced feature numbers reaching the lowest EERs in the time and frequency domain over different time periods in Table 3. As shown in Table 3, the balanced feature number $(29, 21)$ for 2s time period and $(28, 22)$ for 5s time period reach their lowest EERs, respectively.

Considering the balanced feature numbers and the time periods, we concatenate $DeF_t[28]$ time-domain features and $DeF_f[22]$ frequency-domain features as the fused deep feature.

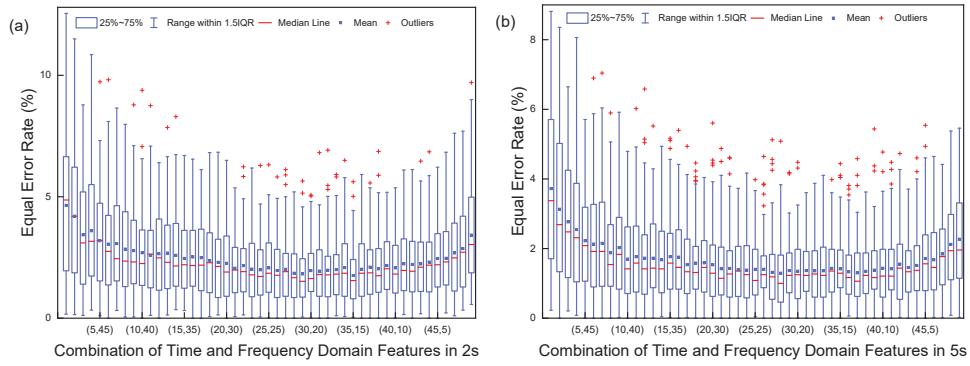

Fig. 6. EER for DeFFusion on combination of time and frequency features in different time periods (2 seconds or 5 seconds)

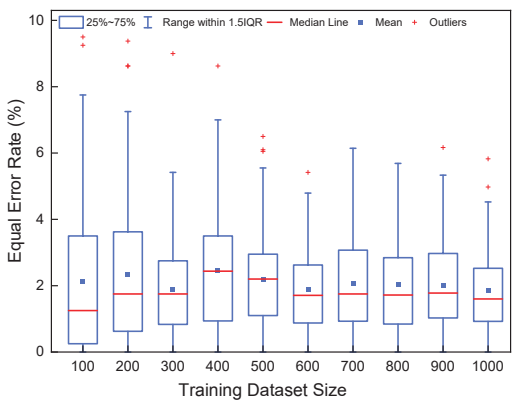

Fig. 7. Accuracy on different training dataset sizes

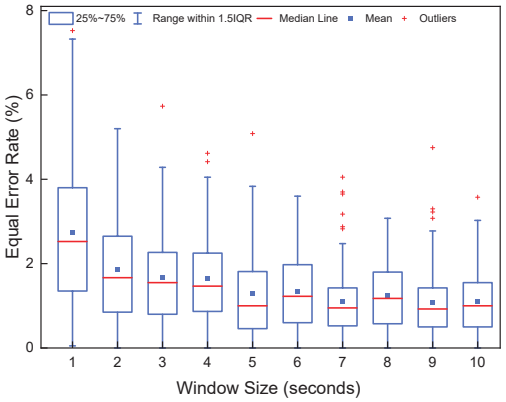

Fig. 8. Accuracy on different time window sizes

5 AUTHENTICATION WITH OC-SVM

With the fused deep features by balanced feature concatenation, DeFFusion utilizes the one-class support vector machine (OC-SVM) classifier to authenticate users. The OC-SVM maps data points into a high-dimensional feature space with a kernel function and finds the surface of a minimal hyper-sphere which contains the objective data points as many as possible [38, 39]. The distance between data points and hyper-sphere is the classification score [40, 41]. In the enrollment phase, the OC-SVM is trained by fused training deep features with the radial basis function kernel, and thus DeFFusion learns the legitimate user's profile from the training data. In the continuous authentication phase, the trained OC-SVM classifies the fused testing deep features. Based on the trained OC-SVM and testing data, DeFFusion classifies the current user as a legitimate user or an impostor. If the user is classified as an impostor, DeFFusion will require initial login inputs; Otherwise, it will allow the continuous usage of the smartphone and meanwhile continuously authenticate the user.

Table 4. Mean, Median and SD of EER (%) on Different Training Dataset Sizes.

Statistics	100	200	300	400	500	600	700	800	900	1000
Mean	2.13	2.34	1.87	2.45	2.20	1.89	2.08	2.03	2.01	**1.86**
Median	1.25	1.75	1.75	2.43	2.20	1.71	1.75	1.72	1.78	**1.60**
SD	2.35	2.19	1.46	1.75	1.42	1.27	1.46	1.44	1.29	**1.29**

6 PERFORMANCE EVALUATION

In this section, we evaluate the performance of DeFFusion. We begin with the experimental settings including dataset collection, CNN and classifier training, and evaluation metrics. Then, we explore the impact of training data size and time window size on classification accuracy. Next, we compare the authentication accuracy on different features over different classifiers and on different classifiers based on the same CNN-extracted features, respectively. Finally, we evaluate the accuracy on unseen users and time efficiency of DeFFusion, and compare DeFFusion with representative authentication methods.

6.1 Experimental Settings

In this section, we report the collected dataset, then describe the training process of the CNN and classifier, and present the metrics for measuring the performance, sequentially.

6.1.1 Dataset. We collect user motion data (accelerometer and gyroscope data) while the users start operating on the phone after login. We developed a data collection tool for Android phones to record the real-time behavioral data invoked by users' interaction with the phones. Data were collected by 100 participants (53 male, and 47 female) using the phones equipped with the developed tool. The participants were asked to conduct three designed tasks: (1) document reading; (2) text production; (3) navigation on a map to locate a destination. When the participants logged into the developed tool, a reading, writing, or map navigation session were randomly assigned, each of which lasted 5 to 15 minutes. Based on the assignments, they were expected to perform 24 sessions (8 reading sessions, 8 writing sessions, and 8 map navigation sessions) with totally 2 to 6 hours of behavior traits.

We select sensor readings of the accelerometer and gyroscope from 95 participants with the sampling rate $f_s = 100\ Hz$ in CSV files on the phones and choose the first 100 minutes of the data for each user with 5-second window sizes as the experimental dataset.

6.1.2 CNN and Classifier Training. For *CNN training*, we select training data with batch size of 256 from all the training data until all are selected. For each batch-size training data, we pass them through the designed CNN, calculate the loss using cross entropy from the CNN output, perform back-propagation and update parameters for the learning rate. We utilize Stochastic Gradient Descent (SGD) optimizer to update the learning rate by $LR_{new} = LR_{ini} \times \gamma^{epoch/step\_size}$, where $step\_size = 30$, $\gamma = 0.1$, and LR_{new} and LR_{ini} indicate the updated and initial learning rates, respectively [42]. The learning rate is initially set as 0.001, and then is gradually reduced by 90% for each 50 epochs. The CNN is trained up to 150 epochs for each batch data.

For *OC-SVM training*, we exploit the ten-fold cross-validation on training data. We randomly select one participant from 95 as the legitimate user and the rest 94 as impostors. Then, the positive samples from the legitimate user are equally divided into 10 subsets, 9 of which are used as the training sets and the rest is used as the testing set. Next, the negative samples from all the impostors with the same size to positives are selected and then divided into 10 subsets, one of which is used as the testing set. The above procedures are repeated 10 times until each subset of positive or negative

Table 5. Mean, Median and SD of EER (%) on Different Time Window Sizes.

Statistics	1s	2s	3s	4s	5s	6s	7s	8s	9s	10s
Mean	2.74	1.85	1.68	1.64	**1.29**	1.35	1.11	1.23	1.08	1.11
Median	2.53	1.67	1.55	1.47	**1.00**	1.23	0.95	1.18	0.93	1.00
SD	1.68	1.18	1.08	1.05	**1.03**	0.87	0.87	0.77	0.84	0.79

samples are tested exactly once. Finally, the ten-fold cross validation training process is repeated 20 times to mitigate the randomness.

6.1.3 Evaluation Metrics. We exploit three representative metrics to evaluate the effectiveness of DeFFusion: false acceptance rate (FAR), false rejection rate (FRR), and equal error rate (EER). We begin with four basic metrics which are used to define the representative metrics: True positive (TP) indicates that operation behaviors from legitimate users are correctly identified; True negative (TN) indicates that operation behaviors not from legitimate users are correctly declined; False positive (FP) indicates that operation behaviors not from legitimate users are incorrectly identified as legitimate; False negative (FN) indicates that operation behaviors from legitimate users are incorrectly rejected. Then, the FAR is the probability that an impostor is falsely classified as a legitimate user, defined as $FAR = \frac{FP}{FP+TN}$ [43–45]. The FRR is the probability that a legitimate user is incorrectly identified as an impostor, defined as $FRR = \frac{FN}{FN+TP}$ [33, 46]. The EER is the point where the FAR equals to the FRR [47–49].

6.2 Impact of Training Dataset Size and Time Window Size

In this section, we evaluate the varying training dataset size and time window size on the authentication accuracy of DeFFusion, respectively.

6.2.1 Impact of Training Dataset Size. The training dataset size impacts the profile of the legitimate user. To investigate the impact of the training dataset sizes, we evaluate DeFFusion authentication accuracy with dataset sizes varying from 100 to 1000 in a step of 100. We demonstrate the box plots of the EER of DeFFusion on different window sizes in Fig. 7. As illustrated in Fig. 7, the mean, median and standard deviation (SD) of EERs slightly fluctuate with the increase of the training dataset size. However, they show a general trend that training with a longer dataset size achieves higher accuracy. Moreover, we tabulate the mean, median and SD of the EER on different training dataset sizes in Table 4. The training dataset size of 1000 reaches the lowest mean EER of 1.86% with a lower 1.29% SD. For the CNN and OC-SVM training, we select 1000 as the training dataset size.

6.2.2 Impact of Time Window Size. The time window size has a significant impact on the classifier training. We evaluate the impact of the time window size on DeFFusion authentication accuracy, with sizes ranging from 1 second to 10 seconds. For each time window size, based on the 95 participants, we utilize the ten-fold cross-validation to train the OC-SVM classifier to obtain the authentication accuracy of DeFFusion. Fig. 8 describes the box plots of the EER of DeFFusion on different window sizes. As demonstrated in Fig. 8, the mean EER (blue solid square) gradually decreases as the time window size increases, which indicates that the more data are trained, the higher accuracy can be achieved. Moreover, the median EER gradually decreases as the time window size increases until the 5 seconds, and then demonstrates slight fluctuations from the 6 to 10 seconds. In addition, Table 5 lists the mean, median and SD of the EER on different time window sizes. As depicted in Table 5, the SD has the same trend to the mean EER, which implies the mean EER tends to be stable as the window size increases. Specifically, DeFFusion achieves 1.29% mean, 1.00%

median and 1.03% SD of the EER, respectively, on the time window size of 5 seconds. Based on the statistics in Table 5, we select the time window size of 5 seconds in the following experiments.

6.3 Accuracy Comparison on Different Features

To evaluate the efficiency of the CNN-extracted features, we compare the authentication accuracy of DeFFusion to comparative schemes exploiting the designed features on representative classifiers, such as OC-SVM [50], k-Nearest Neighbors (kNN) [51], Random Forest (RF) [52], and Decision Tree (DT) [53].

We first introduce the representative classifiers used in comparative schemes:

• *OC-SVM* is an unsupervised learning algorithm, which projects data onto a high-dimensional space through a kernel function and regards the origin as the only sample from other classes [25].

• *kNN* identifies the k training observations that are nearest to the new observation, and selects the label that the majority of the k closest training observations have. It takes every new observation and locates it in feature space with respect to all training observations [51].

• *RF* is a combination of tree-structured predictors such that each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest [52].

• *DT* is a tree-like structure, where leaves represent outcome labels and branches indicate conjunctions of input features that resulted in those outcomes. It separates the data (parent node) into two subsets (child nodes) by calculating the best feature split determined by a chosen split criterion [53].

For the comparative schemes, we specially design $90 = 15 \times 3 \times 2$ features (15 designed features for each axis, 3 axes for each sensor, 2 sensors), where the 15 features for each axis are listed in Table 6. As illustrated in Fig. 1, based on the same datasets, we just replace the CNN-extracted features by the designed features, but the feature selection and fusion processes remain. With the fused features, we utilize the representative classifiers of the OC-SVM, kNN, RF, and DT to conduct the user authentication. Fig. 9 demonstrates the box plots of the EER, FAR and FRR for DeFFusion, comparative schemes with classifiers OC-SVM, kNN, RF, and DT on the designed features, respectively. As shown in Fig. 9, DeFFusion evidently performs the best in the EER, FAR and FRR comparing to the comparative schemes. Moreover, Table 7 lists the EER, FAR and FRR with SD on DeFFusion and representative schemes with different classifiers on designed features. As listed in Table 7, DeFFusion achieves the best authentication accuracy with 1.00% EER, 1.42% FAR and 0.75% FRR, which surpasses the representative schemes with margins of 8.00%, 7.54% and 8.29% at least for the EER, FAR and FRR, respectively. Furthermore, the OC-SVM classifier with designed features shows the best accuracy among the representative schemes, approximately reaching 9.00% EER, 8.96% FAR and 9.04% FRR.

6.4 Accuracy Comparison on Different Classifiers

To evaluate the efficiency of the OC-SVM classifier, we compare the accuracy of DeFFusion with other representative classifiers, such as kNN, RF, and DT. That is, based on the architecture of DeFFusion (Fig. 1), we just replace the OC-SVM classifier with the classifiers of kNN, RF and DT, respectively, but other modules remain. Based on the same datasets, we calculate the authentication accuracy using classifiers of kNN, RF and DT, respectively. Fig. 10 demonstrates the box plots of EER, FAR and FRR for DeFFusion with different classifiers on the same CNN-extracted features, and the corresponding results for these classifiers are listed in Table 8. Since we use the fused deep features extracted by CNN for the classifiers for comparison, we denote OC-SVM classifier by DeFFusion, kNN by CNN-kNN, RF by CNN-RF, and DT by CNN-DT, respectively. As shown in Fig. 10 and Table 8, all the classifiers obviously perform better when using CNN-extracted features (compared

Table 6. Designed Features.

Feature	Explanation
Mean	Mean value of one-axis sensor readings
SD	Standard deviation of one-axis sensor readings
Maximum	Maximum value of one-axis sensor readings
Minimum	Minimum value of one-axis sensor readings
Range	Difference between the maximum and minimum values
Kurtosis	Width of peak of one-axis sensor readings
Skewness	Orientation of peak of one-axis sensor readings
Quartiles	25%, 50%, 75% quartiles of one-axis readings
Energy	Intensity of one axis sensor readings
Entropy	Dispersion of spectral distribution of one-axis readings
P1	Amplitude of the 1st highest peak of one-axis readings
P2F	Frequency of the 2nd highest peak of one-axis readings
P2	Amplitude of the 2nd highest peak of one-axis readings

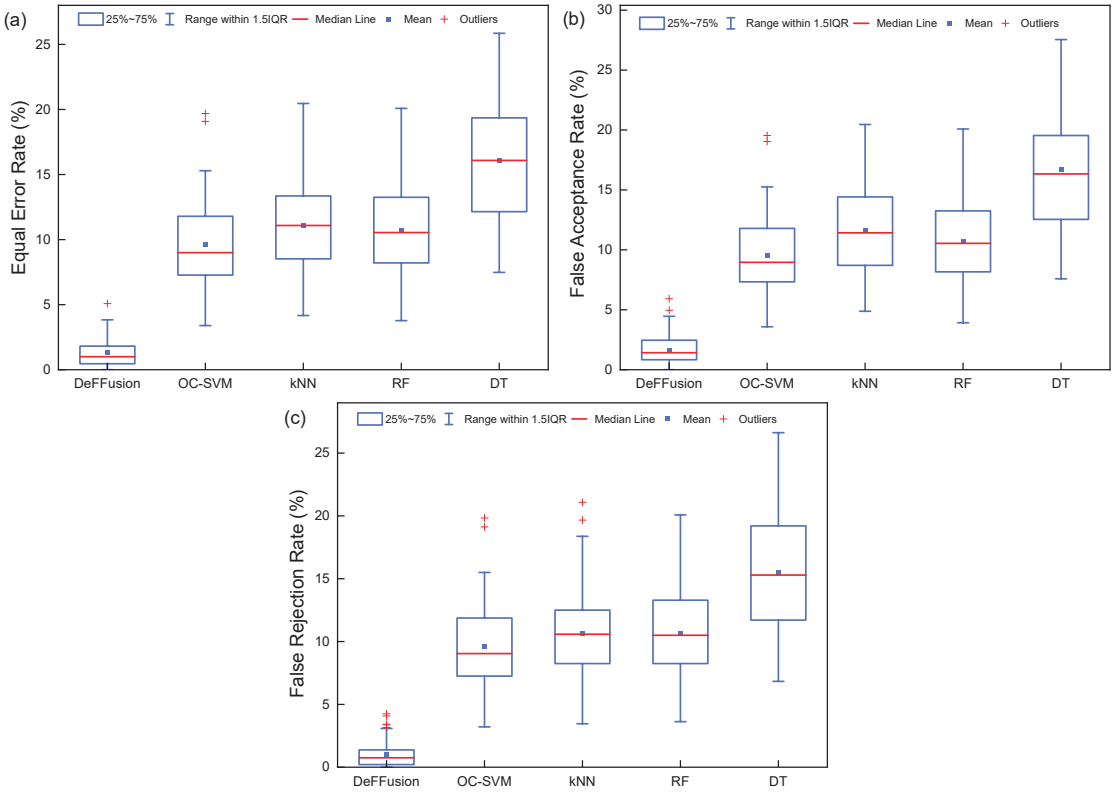

Fig. 9. EER, FAR and FRR for DeFFusion on different features with different classifiers.

Table 7. EER, FAR, and FRR (%) with SD on Different Features

Scheme	EER (SD)	FAR (SD)	FRR (SD)
DeFFusion	1.00 (1.03)	1.42 (1.26)	0.75 (0.99)
OC-SVM	9.00 (3.22)	8.96 (3.21)	9.04 (3.24)
kNN	11.08 (3.19)	11.42 (3.38)	10.58 (3.22)
RF	10.54 (3.49)	10.54 (3.46)	10.50 (3.53)
DT	16.08 (4.72)	16.33 (4.81)	15.29 (4.86)

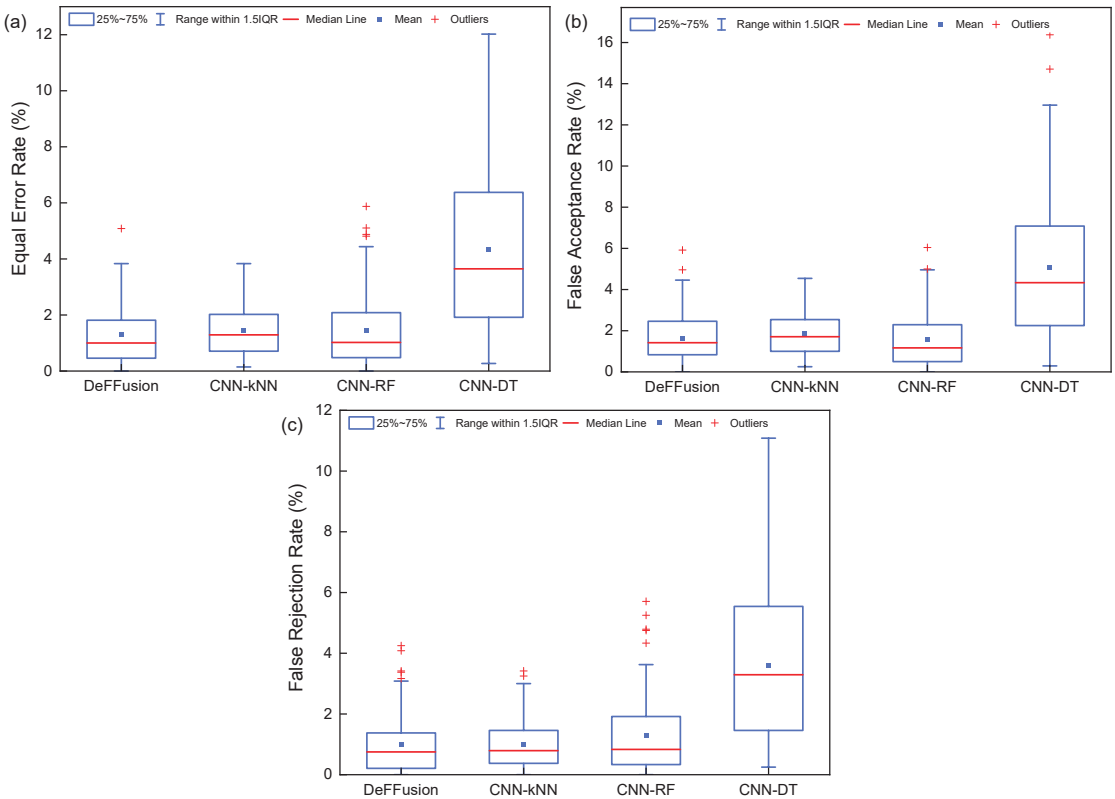

Fig. 10. EER, FAR and FRR for DeFFusion on different classifiers.

with Fig. 9), where DT demonstrates the most improvements, with the margins of 12.43%, 12.00% and 12.00% for the EER, FAR and FRR, respectively (compared with Table 7). Then, DeFFusion outperforms the other three classifiers in the EER and FRR, with margins of 0.02% and 0.04% at least, respectively. CNN-RF shows a slightly better FAR of 1.17%, but the corresponding SD of 1.31% is slightly higher than that of 1.26% for DeFFusion.

6.5 Accuracy on Unseen Users

To evaluate the performance of OC-SVM classifier on unseen users, we calculate the DeFFusion accuracy with different pre-trained classifiers. With the collected dataset consisting of 95 users, we randomly select $(95 - m)$ users (unseen users) to train the OC-SVM classifier and the rest m users

Table 8. EER, FAR, and FRR (%) with SD on Different Classifiers

Classifier	EER (SD)	FAR (SD)	FRR (SD)
DeFFusion	1.00 (1.03)	1.42 (1.26)	0.75 (0.99)
CNN-kNN	1.29 (0.92)	1.71 (1.08)	0.79 (0.81)
CNN-RF	1.02 (1.29)	1.17 (1.31)	0.83 (1.30)
CNN-DT	3.65 (2.86)	4.33 (3.40)	3.29 (2.46)

Table 9. EER, FAR and FRR (%) with SD on Different Number of Unseen Users.

Unseen User	20	30	40	50	60	70
EER(SD)	4.23 (1.87)	2.40 (1.54)	2.38 (1.45)	2.15 (1.17)	1.83 (0.96)	1.92 (1.06)
FAR(SD)	4.00 (1.87)	2.71 (1.54)	2.92 (1.49)	2.92 (1.57)	3.25 (1.46)	3.31 (1.68)
FRR(SD)	4.25 (1.96)	2.46 (1.74)	1.83 (1.57)	1.04 (1.01)	0.63 (0.95)	0.33 (0.82)

are used to test it. For classifier training, we also exploit the ten-fold cross-validation on training data of $(95 - m)$ users, where one of $(95 - m)$ users is randomly selected as the legitimate user and the rest $(95 - m - 1)$ are used as impostors. To generalize the DeFFusion accuracy, we assign the number of unseen users as $(95 - m) = 20, 30, 40, 50, 60, 70$, respectively. We present the box plots of the EER, FAR, and FRR for DeFFusion on different number of unseen users in Fig. 11. As illustrated in Fig. 11(a), the EER gradually decreases as the increase of the unseen user number until 60 and then slightly increases on 70. The FAR in Fig. 11(b) gradually rises as the growth of the unseen users from 30 and the FRR generally decreases as the unseen users grow in Fig. 11(c). In addition, Table 9 lists the mean EER, FAR, FRR with SD on different number of unseen users. As depicted in Table 9, when we select 60 unseen users to train the OC-SVM classifier and the rest 35 to test it, DeFFusion achieves the best accuracy with 1.83% EER, 3.25% FAR, and 0.63% FRR. However, with 20 unseen users, DeFFusion receives the lowest accuracy with 4.23% EER, 4.00% FAR, and 4.25% FRR, respectively.

6.6 Time Efficiency

To evaluate DeFFusion performance in the authentication phase, we compute the time cost for the designed CNN on feature extraction and the OC-SVM classifier on classification, respectively. We deployed DeFFusion on Samsung Galaxy S4 smartphones with the trained CNN and OC-SVM classifier. Note that the CNN extractor and OC-SVM classifier are trained in the enrollment phase. After setting up the DeFFusion parameters of 5-second window size and 1000 data size, we measure the average time for DeFFusion conducting an authentication to be less than 20 ms, where the designed CNN on feature extraction spends roughly 15.2 ms and the OC-SVM consumed approximately 3 ms. Therefore, considering the window size of 5 seconds, the time for DeFFusion executing a continuous authentication is roughly 5 seconds.

6.7 Comparison with Representative Authentication Methods

To compare DeFFusion with the state-of-the-art authentication methods: FinAuth [40], SCANet [25], HMOG [54], and Multi-Motion [38], we analyze the difference between DeFFusion and these representative methods from the aspects of sensor source, feature extraction methods, participants' number in experiments, classifiers, and authentication performance in terms of the accuracy and time, as illustrated in Table 10. FinAuth and SCANet utilize CNN-extracted features (without feature fusion) to train local outlier factor (LOF) and OC-SVM classifiers, but both achieve lower

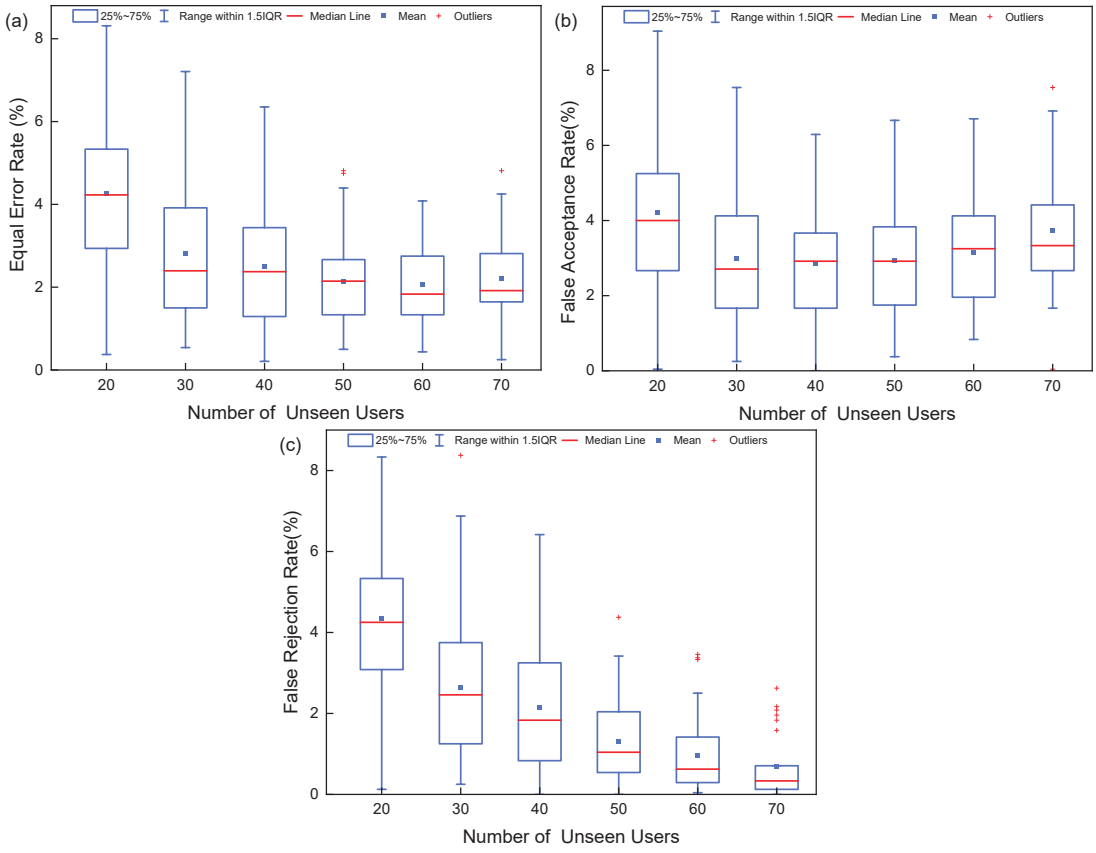

Fig. 11. EER, FAR and FRR for DeFFusion on different number of unseen users.

Table 10. Comparison with Representative Authentication Methods.

Method	Sensor	Feature	Participant	Classifier	Authentication	
					Accuracy	Time
DeFFusion	Acc., Gyr.	CNN-extracted	95	OC-SVM	1.00% EER	~5s
FinAuth	Acc., Gyr., Mag.	CNN-extracted	90	LOF	97.99% BAC	~713ms
SCANet	Acc., Gyr.	CNN-extracted	100	OC-SVM	2.35% EER	~3s
HMOG	Acc., Gyr., Mag.	HMOG	100	Scaled Manhattan	7.16% EER	~60s
Multi-Motion	Acc., Gyr., Mag., Ori.	Descriptive and intensive	102	HMM	4.74% EER	~8s

accuracy of 97.99% BAC and 2.35% EER, respectively. Moreover, HMOG and Multi-Motion exploit the designed features of the hand movement, orientation, and grasp (HMOG), and descriptive and intensive features to train scaled Manhattan and hidden Markov model (HMM) classifiers, and reach lower accuracy (7.16% and 4.74%) and take longer authentication time (60s and 8s), respectively. With the fused CNN-extracted features, DeFFusion achieves the best accuracy of 1.00% EER and approximately 5s time delay.

7 RELATED WORK

In this section, we provide a literature review on deep learning in recognition systems and biometric fusion in recognition systems, respectively.

7.1 Deep Learning in Recognition Systems

Deep learning involves stacking multiple layers of learning algorithms to approximate highly nonlinear functions, which enables deep learning algorithms to learn hierarchical representations/features from data for recognition systems [55].

Deep learning approaches for the various biometric modalities can be broadly categorized into physiological biometrics (e.g., fingerprint [8, 56], face [9, 57], palmprint [58], and iris [26]) and behavioral biometrics (e.g., voice [18], signature [19, 20], gait [14, 21], and keystroke [22]). Specifically, for deep learning in physiological biometrics, the authors in [56] proposed an automated latent fingerprint recognition algorithm that utilized a CNN for ridge flow estimation and minutiae descriptor extraction, and extracted complementary templates to represent the latent. In [57], the authors proposed a multitask, parts-based CNN for estimating attributes to enable continuous mobile device authentication, where deep and wide variations of two CNNs were trained: BinaryCNNs that were trained on a single attribute and MultiCNNs that were trained on multiple attributes. The authors in [58] used a two-layer deep scattering CNN for palmprint recognition, where scattering networks were similar to CNNs, except that they used predefined wavelet transform filters rather than learning filters from data. In [26], the authors used two deep CNN architectures for iris recognition, one with eight convolutional layers and another with five convolution and two inception layers. Although deep learning is effective, these physiological biometrics-based approaches require direct user participation in the process of the authentication. For deep learning in behavioral biometrics, the authors in [18] proposed an integrated deep learning system that provided a verification score given few reference utterances and a test utterance. In [19], the authors proposed an online signature verification framework based on deep convolutional Siamese network, which automatically extracted robust feature descriptions based on metric-based loss function. The authors in [20] exploited RNNs for the sequential nature of online verification by training a two-layer RNN with a length normalized path signature descriptor as input and triplet loss. In [21], the authors proposed to use CNNs and multi-task learning model to identify human gait and to predict multiple human attributes simultaneously. The authors in [22] exploited CNNs to derive discriminative characteristics from the typing patterns of subjects entering personal identification numbers based on CNNs. However, most of these behavioral biometrics-based approaches consisting of many layers and channels require much computational budget.

Different from these representative deep-learning recognition systems, we specially design a light-weight and effective CNN architecture to learn and extract motion sensor features for mobile user authentication without their direct involvement.

7.2 Biometric Fusion in Recognition Systems

Multi-biometric recognition systems utilize the principle of fusion to combine information from multiple sources in order to improve recognition accuracy whilst addressing some of the limitations in single-biometric systems [28].

Biometric fusion strategies can generally be divided into sensor-level fusion [30], feature-level fusion [27, 29], score-level fusion [31], and decision-level fusion [32]. Concretely, in [30], the authors introduced a multimodal biometric system based on face and palmprint fusion with bit-plane decomposition approach. The authors in [27] applied maxout units into the CNNs to generate a compact representation for iris and periocular biometrics in images and then fused the two

modalities of image features through a weighted concatenation for mobile recognition. In [29], the authors used the serial fusion and parallel fusion to directly combine the designed features for user authentication. Inspired by the feature-level fusion, we balance two types of CNN-extracted sensor features by receiving the best accuracy to fuse the features of the accelerometer and gyroscope for continuous authentication in our work. The authors in [31] utilized CNNs to extract pores from raw fingerprint patches to aid automatic fingerprint identification, which were combined with minutiae and ridge patterns extracted using conventional approaches and fused using a unique matching scheme. In [32], the authors proposed a security analysis framework that combined information-theoretic approach with computational security, and constructed a fingerprint-based multibiometric cryptosystem using decision level fusion.

Although biometric fusion strategies have been used in these excellent recognition systems, we differ in that we utilize a balanced feature concatenation to fuse the motion-sensor features in a CNN-based continuous authentication system on smartphones.

8 CONCLUSION AND LIMITATION

In this paper, we propose DeFFusion, a CNN-based continuous authentication system using deep feature fusion for smartphone users, by leveraging the accelerometer and gyroscope ubiquitously built into smartphones. DeFFusion is composed of five modules: data collection, data preprocessing, deep feature extraction, classification, and authentication. Based on the collected data, DeFFusion first converts the time domain data into frequency domain data using the fast Fourier transform and then inputs both of them into a designed CNN, respectively. With the CNN-extracted features, DeFFusion conducts the feature selection utilizing factor analysis and exploits balanced feature concatenation to fuse these deep features. Based on the one-class SVM classifier, DeFFusion authenticates the current users as a legitimate user or an impostor. To validate the authentication performance of DeFFusion, we conduct extensive experiments in terms of impact of training data size and time window size, accuracy comparison on different features and different classifiers, accuracy on unseen users and time efficiency, and comparison with representative authentication methods. The experimental results show that DeFFusion performs the best accuracy on different features (CNN-extracted features vs designed features) with different classifiers (OC-SVM, kNN, RF, and DT) and on different classifiers (kNN, RF, and DT) with the same CNN-extracted features, by achieving 1.00% EER, 1.42% FAR and 0.75% FRR, in 5-second time window size.

Although we take significant efforts to validate the effectiveness of DeFFusion, there are some limitations in our studies and experiments. For example, different holding postures may incur different patterns of motion sensor data, which undermine the usability and robustness of our approach. The dataset we collected was from limited subjects that my cause unbalanced demographic characteristics, such as genders, regions, and ages.

ACKNOWLEDGMENTS

We thank the subjects for collecting the experimental data and appreciate the anonymous reviewers for their valuable suggestions. This work was partially supported by the National Natural Science Foundation of China under Grant 62072061 and by the Fundamental Research Funds for the Central Universities under Grant 2021CDJQY-026.

REFERENCES

[1] Christian Montag, Alexander Markowetz, Konrad Blaszkiewicz, Ionut Andone, Bernd Lachmann, Rayna Sariyska, Boris Trendafilov, Mark Eibes, Julia Kolb, Martin Reuter, et al. Facebook usage on smartphones and gray matter volume of the nucleus accumbens. *Behav. Brain Res.*, 329:221–228, 2017.

[2] Mingyang Zhang, Tong Li, Hancheng Cao, Yong Li, Sasu Tarkoma, and Pan Hui. "what apps did you use?": Understanding the long-term evolution of mobile app usage. In *The Web Conference (WWW'20)*, pages 66–76, 2020.
[3] Milad Taleby Ahvanooey, Qianmu Li, Mahdi Rabbani, and Ahmed Raza Rajput. A survey on smartphones security: Software vulnerabilities, malware, and attacks. *Int. J. Adv. Comput. Sci. Appl.*, 8(10), 2017.
[4] Youngho Kim, Tae Oh, and Jeongnyeo Kim. Analyzing user awareness of privacy data leak in mobile applications. *Mob. Inf. Syst*, 2015:50–57, 2015.
[5] Vishal M Patel, Rama Chellappa, Deepak Chandra, and Brandon Barbello. Continuous user authentication on mobile devices: Recent progress and remaining challenges. *IEEE Signal Process. Mag.*, 33(4):49–61, 2016.
[6] Toan Van Nguyen, Napa Sae-Bae, and Nasir Memon. Draw-a-pin: Authentication using finger-drawn pin on touch devices. *Comput. Secur.*, 66:115–128, 2017.
[7] Weizhi Meng, Liqiu Zhu, Wenjuan Li, Jinguang Han, and Yan Li. Enhancing the security of fintech applications with map-based graphical password authentication. *Future Gener. Comput. Syst.*, 101:1018–1027, 2019.
[8] Xinchen Zhang, Yafeng Yin, Lei Xie, Hao Zhang, Zefan Ge, and Sanglu Lu. Touchid: User authentication on mobile devices via inertial-touch gesture analysis. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, 4(4), December 2020.
[9] Stepan Komkov and Aleksandr Petiushko. Advhat: Real-world adversarial attack on arcface face id system. In *2020 25th International Conference on Pattern Recognition (ICPR'20)*, pages 819–826, 2021.
[10] Adam J. Aviv, Katherine Gibson, Evan Mossop, Matt Blaze, and Jonathan M. Smith. Smudge attacks on smartphone touch screens. In *the 4th USENIX Conference on Offensive Technologies (WOOT'10)*, page 1–7, 2010.
[11] Upal Mahbub, Vishal M Patel, Deepak Chandra, Brandon Barbello, and Rama Chellappa. Partial face detection for continuous authentication. In *2016 IEEE International Conference on Image Processing (ICIP'16)*, pages 2991–2995. IEEE, 2016.
[12] Niinuma Kawasaki, Unsang Park, and Anil K. Jain. Soft biometric traits for continuous user authentication. *IEEE Trans. Inf. Forens. Secur.*, 5(4):771–780, 2010.
[13] Ge Peng, Gang Zhou, David T. Nguyen, Xin Qi, Qing Yang, and Shuangquan Wang. Continuous authentication with touch behavioral biometrics and voice on wearable glasses. *IEEE Trans. Hum. Mach. Syst.*, 47(3):404–416, 2017.
[14] Muhammad Muaaz and René Mayrhofer. Smartphone-based gait recognition: From authentication to imitation. *IEEE Trans. Mob. Comput.*, 16(11):3209–3221, 2017.
[15] Yong Jin, Masahiko Tomoishi, and Satoshi Matsuura. An in-depth concealed file system with gps authentication adaptable for multiple locations. In *2017 IEEE 41st Annual Computer Software and Applications Conference (COMPSAC'17)*, volume 1, pages 608–613, 2017.
[16] Hailong Hu, Yantao Li, Zhangqian Zhu, and Gang Zhou. Cnnauth: Continuous authentication via two-stream convolutional neural networks. In *2018 IEEE International Conference on Networking, Architecture and Storage (NAS'18)*, pages 1–9. IEEE, 2018.
[17] Mohammed Abuhamad, Tamer Abuhmed, David Mohaisen, and Dae Hun Nyang. Autosen: Deep-learning-based implicit continuous authentication using smartphone sensors. *IEEE Internet Things J.*, 7(6):5008–5020, 2020.
[18] Georg Heigold, Ignacio Moreno, Samy Bengio, and Noam Shazeer. End-to-end text-dependent speaker verification. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'16)*, pages 5115–5119, 2016.
[19] Chandra Sekhar Vorugunti, Guru Devanur S., Prerana Mukherjee, and Viswanath Pulabaigari. Osvnet: Convolutional siamese network for writer independent online signature verification. In *2019 International Conference on Document Analysis and Recognition (ICDAR'19)*, pages 1470–1475, 2019.
[20] Songxuan Lai, Lianwen Jin, and Weixin Yang. Online signature verification using recurrent neural network and length-normalized path signature descriptor. In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR'17)*, volume 01, pages 400–405, 2017.
[21] Chao Yan, Bailing Zhang, and Frans Coenen. Multi-attributes gait identification by convolutional neural networks. In *2015 8th International Congress on Image and Signal Processing (CISP'15)*, pages 642–647, 2015.
[22] Emanuele Maiorana, Himanka Kalita, and Patrizio Campisi. Deepkey: Keystroke dynamics and cnn for biometric recognition on mobile devices. In *2019 8th European Workshop on Visual Information Processing (EUVIP'19)*, pages 181–186, 2019.
[23] Mario Parreno Centeno, Aad van Moorsel, and Stefano Castruccio. Smartphone continuous authentication using deep learning autoencoders. In *2017 15th Annual Conference on Privacy, Security and Trust (PST'17)*, pages 147–1478. IEEE, 2017.
[24] Chris Xiaoxuan Lu, Bowen Du, Peijun Zhao, Hongkai Wen, Yiran Shen, Andrew Markham, and Niki Trigoni. Deepauth: in-situ authentication for smartwatches via deeply learned behavioural biometrics. In *2018 ACM International Symposium on Wearable Computers (ISWC'18)*, pages 204–207, 2018.
[25] Yantao Li, Hailong Hu, Zhangqian Zhu, and Gang Zhou. Scanet: Sensor-based continuous authentication with two-stream convolutional neural networks. *ACM Trans. Sen. Netw.*, 16(3), July 2020.

[26] Abhishek Gangwar and Akanksha Joshi. Deepirisnet: Deep iris representation with applications in iris recognition and cross-sensor iris recognition. In *2016 IEEE International Conference on Image Processing (ICIP'16)*, pages 2301–2305, 2016.

[27] Qi Zhang, Haiqing Li, Zhenan Sun, and Tieniu Tan. Deep feature fusion for iris and periocular biometrics on mobile devices. *IEEE Trans. Inf. Forens. Secur.*, 13(11):2897–2912, 2018.

[28] Maneet Singha, Richa Singha, and Arun Rossb. A comprehensive overview of biometric fusion. *Inf. Fusion*, 52:187–205, 2019.

[29] Yantao Li, Bin Zou, Shaojiang Deng, and Gang Zhou. Using feature fusion strategies in continuous authentication on smartphones. *IEEE Internet Comput.*, 24(2):49–56, 2020.

[30] Therry Z. Lee and David B. L. Bong. Face and palmprint multimodal biometric system based on bit-plane decomposition approach. In *2016 IEEE International Conference on Consumer Electronics-Taiwan (ICCE-TW'16)*, pages 1–2, 2016.

[31] Hong-Ren Su, Kuang-Yu Chen, Wei Jing Wong, and Shang-Hong Lai. A deep learning approach towards pore extraction for high-resolution fingerprint recognition. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'17)*, pages 2057–2061, 2017.

[32] Cai Li, Jiankun Hu, Josef Pieprzyk, and Willy Susilo. A new biocryptosystem-oriented security analysis framework and implementation of multibiometric cryptosystems based on decision level fusion. *IEEE Trans. Inf. Forens. Secur.*, 10(6):1193–1206, 2015.

[33] Mohammed Abuhamad, Ahmed Abusnaina, DaeHun Nyang, and David Mohaisen. Sensor-based continuous authentication of smartphones' users using behavioral biometrics: A contemporary survey. *IEEE Internet Things J.*, 8(1):65–84, 2021.

[34] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR'18)*, pages 6848–6856, 2018.

[35] Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *The 15th European Conference on Computer Vision (ECCV'18)*, pages 122–138, Cham, 2018. Springer International Publishing.

[36] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'16)*, pages 770–778, 2016.

[37] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR'18)*, pages 4510–4520, 2018.

[38] Chao Shen, Yuanxun Li, Yufei Chen, Xiaohong Guan, and Roy A. Maxion. Performance analysis of multi-motion sensor behavior for active smartphone authentication. *IEEE Trans. Inf. Forens. Secur.*, 13(1):48–62, 2018.

[39] Chao Shen, Tianwen Yu, Sheng Yuan, Yunpeng Li, and Xiaohong Guan. Performance analysis of motion-sensor behavior for user authentication on smartphones. *Sensors*, 16(3):345, 2016.

[40] Cong Wu, Kun He, Jing Chen, Ziming Zhao, and Ruiying Du. Liveness is not enough: Enhancing fingerprint authentication with behavioral biometrics to defeat puppet attacks. In *29th USENIX Security Symposium (USENIX Security'20)*, pages 2219–2236. USENIX Association, August 2020.

[41] Alexander Senf, Xue-wen Chen, and Anne Zhang. Comparison of one-class svm and two-class svm for fold recognition. In *International Conference on Neural Information Processing (ICONIP'06)*, pages 140–149. Springer, 2006.

[42] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *19th International Conference on Computational Statistics (COMPSTAT'10)*, pages 177–186. Springer, 2010.

[43] Pei-Yuan Wu, Chi-Chen Fang, Jien Morris Chang, and Sun-Yuan Kung. Cost-effective kernel ridge regression implementation for keystroke-based active authentication system. *IEEE Trans. Cybernet.*, 47(11):3916–3927, 2016.

[44] Mohsen Ali Alawami, William Aiken, and Hyoungshick Kim. Lightlock: user identification system using light intensity readings on smartphones. *IEEE Sens. J.*, 20(5):2710–2721, 2019.

[45] Xiangmao Chang, Cheng Peng, Guoliang Xing, Tian Hao, and Gang Zhou. Isleep: A smartphone system for unobtrusive sleep quality monitoring. *ACM Trans. Sen. Netw.*, 16(3), July 2020.

[46] Lingjun Li, Xinxin Zhao, and Guoliang Xue. Unobservable re-authentication for smartphones. In *20th Annual Network & Distributed System Security Symposium (NDSS'13)*, volume 56, pages 57–59, 2013.

[47] Arsalan Mosenia, Susmita Sur-Kolay, Anand Raghunathan, and Niraj K Jha. Caba: Continuous authentication based on bioaura. *IEEE Trans. Comput.*, 66(5):759–772, 2016.

[48] Shuochao Yao, Shaohan Hu, Yiran Zhao, Aston Zhang, and Tarek Abdelzaher. Deepsense: A unified deep learning framework for time-series mobile sensing data processing. In *26th International Conference on World Wide Web (WWW'17)*, pages 351–360, 2017.

[49] Ivan Martinovic, Kasper Rasmussen, Marc Roeschlin, and Gene Tsudik. Authentication using pulse-response biometrics. *Commun. ACM*, 60(2):108–115, 2017.

[50] Huan Feng, Kassem Fawaz, and Kang G Shin. Continuous authentication for voice assistants. In *23rd Annual International Conference on Mobile Computing and Networking (MobiCom'17)*, pages 343–355, 2017.
[51] L. E. Peterson. K-nearest neighbor. *Scholarpedia*, 4(2):1883, 2009.
[52] Leo Breiman. Random forests. *Mach. Learn.*, 45(1):5–32, 2001.
[53] Torgyn Shaikhina, Dave Lowe, Sunil Daga, David Briggs, Robert Higgins, and Natasha Khovanova. Decision tree and random forest models for outcome prediction in antibody incompatible kidney transplantation. *Biomed. Signal Process.*, 52:456–462, 2019.
[54] Zdeňka Sitová, Jaroslav Šeděnka, Qing Yang, Ge Peng, Gang Zhou, Paolo Gasti, and Kiran S. Balagani. Hmog: New behavioral biometric features for continuous authentication of smartphone users. *IEEE Trans. Inf. Forens. Secur.*, 11(5):877–892, 2016.
[55] Kalaivani Sundararajan and Damon L. Woodard. Deep learning for biometrics: A survey. *ACM Comput. Surv.*, 51(3), May 2018.
[56] Kai Cao and Anil K. Jain. Automated latent fingerprint recognition. *IEEE Trans. Pattern Anal. Mach. Intell. 20*, 41(4):788–800, 2019.
[57] Pouya Samangouei and Rama Chellappa. Convolutional neural networks for attribute-based active authentication on mobile devices. In *2016 IEEE 8th International Conference on Biometrics Theory, Applications and Systems (BTAS'16)*, pages 1–8, 2016.
[58] Shrevin Minaee and Yao Wang. Palmprint recognition using deep scattering network. In *2017 IEEE International Symposium on Circuits and Systems (ISCAS'17)*, pages 1–4, 2017.