TrapCog: An Anti-noise, Transferable, and Privacy-preserving Real-time Mobile User Authentication System with High Accuracy

Jie Ying, Tiantian Zhu*, Qiang Liu, Chunlin Xiong, Zhengqiu Weng, Tieming Chen, Lei Fu, *Member, IEEE*, Mingqi Lv, Han Wu, Ting Wang, Yan Chen, *Fellow, IEEE*

Abstract—The authentication technology of mobile device users has been studied for decades. To balance security, privacy, and usability, motion sensors-based user authentication methods are widely investigated in recent years. However, existing studies meet the problems such as scarcity of training samples, underutilization of data, poor de-noising ability, insufficient transferability, privacy leakage, and low accuracy. To overcome these difficulties, we propose a system, called TrapCog, with the following capabilities: 1) In the phase of data collection, TrapCog can eliminate man-made noise (mislabeling) through differential training based on down-sampling. 2) In the model training stage, the siamese neural network with Long Short-Term Memory (LSTM) as the sub-network is used to achieve sufficient coverage of sample patterns and the transferability of the model. 3) In the phase of real-world authentication, the privacy of the user is tremendously protected through end-side model deployment and local authentication. Experimental results on a dataset composed of 1,513 users with real-world noise show that TrapCog has high accuracy and strong transferability, which is much better than state-of-the-art studies.

Index Terms—User Authentication, Mobile Device, Differential Training, Deep Learning.

1 Introduction

The rapid growth of storage and computation ability makes the mobile device a valuable tool for Internet activities. Nowadays mobile devices (especially smartphones) have become one of the main platforms for users to communicate and interact with different forms of data and media.

To prevent user information from being leaked, various technologies have been proposed to authenticate the mobile user. According to different objects, the authentication methods can be divided into two main categories: knowledge-based and biometric-based. Knowledge-based methods require users to provide specific information (e.g., passwords, PINs, and graphical gestures [15]) for subsequent access [16]. Although such methods are inexpensive, they still suffer from the inconvenience of, e.g., repeated input in a smaller dialog box, and from several representative attacks (e.g., brute-force attacks, shoulder surfing attacks, touchscreen smudges, and sensor-based inferring [17–22]). Correspondingly, authentication methods based on biometrics (e.g., static biometrics such as faces and fingerprints) are widely accepted by users due to their higher efficiency and accuracy [23–25]. However, the aforementioned static biometric technologies require the user to participate in the authentication process explicitly. For example, the user must face the camera or move the finger to the fingerprint sensor. Frequent human-device interaction will undoubtedly reduce the user experience. In addition, the collection of biological content will also cause users to worry about the leakage of their personal privacy information.

Due to the increased user requirements for the security, usability, and privacy of the authentication system, an authentication system that is user-friendly, adaptable to different scenarios, of high accuracy, and privacy-preserving is urgently required. In recent years, motion sensors-based dynamic user authentication studies are widely investigated [1–14]. These methods usually collect data from a series of motion sensors such as acceleration sensors, gravity sensors, and gyroscope sensors. By adopting various machine learning or deep learning algorithms, the unique behavior patterns of the user's gait or gesture are identified, to achieve the purpose of user authentication. Among these studies, the most representative work is ESPIALCOG [1], which implements a general, efficient and robust mobile user implicit authentication system. However, the problems such as low coverage of human-device interaction patterns, weak denoising ability, insufficient transferability, privacy leakage, and low accuracy still exist, rendering it difficult for motion sensors-based dynamic user authentication technology to be practical in the real-world scenarios.

To deal with the above problems, in this paper, we design an anti-noise, high coverage, transferable, and privacy-

- *This work is supported in part by the following grants: National Natural Science Foundation of China under Grant No. 62002324, No. 52005442, No.U22B2028, and No. U1936215. Fuxi Foundation of CCF-SANGFOR under Grant No. 20220201. Zhejiang Provincial Natural Science Foundation of China under Grant No. LQ21F020016, LY22E050014 and LY20F020027. Key R&D Projects in Zhejiang Province under Grant No. 2021C01117. "Ten Thousand People Program" Technology Innovation Leading Talent Project in Zhejiang Province under Grant No. 2020R52011; Major Program of Natural Science Foundation of Zhejiang Province under Grant No. LD22F020002.*
- *Jie Ying, Tiantian Zhu*, Tieming Chen, Mingqi Lv, Han Wu, and Ting Wang are with the College of Computer Science and Technology, Zhejiang University of Technology, Hangzhou 310023, China. E-mail: jieying@zjut.edu.cn, ttzhu@zjut.edu.cn, tmchen@zjut.edu.cn, mingqilv@zjut.edu.cn, if.far.away@ahnu.edu.cn, wangting@zjut.edu.cn. *corresponding author*
- *Lei Fu is with the College of Mechanical Engineering, Zhejiang University of Technology, Hangzhou 310023, China. E-mail: fulei@zjut.edu.cn.*
- *Chunlin Xiong is with Department of Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen 518052, China. E-mail: xiongchunlin@sangfor.com.cn.*
- *Zhengqiu Weng is with School of Wenzhou University of Technology, Wenzhou, Zhejiang 325035, China. E-mail: derisweng@qq.com.*
- *Qiang Liu is with the College of Computer Science and Technology, Zhejiang University, Hangzhou 310027, China. E-mail: qiangliu@zju.edu.cn.*
- *Yan Chen is with Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL 60208, USA. E-mail: ychen@northwestern.edu.*

TABLE 1
Comparison with related studies on smartphone sensor-based authentication. L represents low, M represents medium, and H represents high. NA represents the information is not mentioned.

Study	Require User Movement	Device Placement	Latency	Interaction Coverage	De-Noise Ability	Transferable	Accuracy
TRAPCOG	No	Dynamic	L	H	H	Yes	H
ESPIALCOG(2020) [1]	No	Dynamic	L	M	M	No	M
RISKCOG (2019) [2]	No	Dynamic	L	M	L	No	M
Derawi et al. (2010) [3]	Yes	Fixed	NA	L	NA	No	L
Kwapisz et al. (2010) [4]	Yes	Fixed	NA	L	NA	No	M
Ho et al. (2012) [5]	Yes	Fixed	NA	L	NA	No	NA
Zhu et al. (2013) [6]	No	Fixed	H	L	NA	No	L
Lu et al. (2014) [7]	Yes	Dynamic	M	L	NA	No	M
Kayacik et al (2014) [8]	Yes	Fixed	H	L	NA	No	M
Ren et al. (2015) [9]	Yes	Fixed	NA	L	NA	No	M
Lee et al. (2015) [10]	Yes	Fixed	M	L	NA	No	M
Sitová et al. (2016) [11]	Yes	Fixed	M	L	NA	No	M
Lee et al. (2017) [12]	No	Fixed	L	L	NA	No	M
Buriro et al. (2017) [13]	Yes	Dynamic	NA	L	NA	No	M
Shen et al. (2018) [14]	Yes	Fixed	M	L	NA	No	M

preserving real-time mobile user authentication system based on motion sensors, called TRAPCOG. In Table 1, we list the problems of previous motion sensor-based methods and summarize four challenges as follows:

(1) Weak de-noising ability. Almost all previous study [1–14] about user authentication based on dynamic biometirc relied on training machine learning model. However, the construction of the model is closely related to the quality of the dataset, and the noise in it will greatly affect the quality of the model. For data collected in a non-lab environment, noise can be divided into machine abnormality and man-made noise. From Table 1, we can find that most of the existing studies hardly considered the influence of noise except RISKCOG [2] and ESPIALCOG [1]. The former was able to remove the flat data which was not related to the behaviors of users. The latter eliminated three anomalies caused by the machine/hardware. However, in the real-world scenario, the specific behaviors of users during the data collection period cannot be determined (i.e., temporarily lending smartphones to others). These behaviors will undoubtedly cause mislabeling and affect the accuracy of the final authentication. For this problem, RISKCOG took a semi-supervised learning method to reduce noisy labels and ESPIALCOG took an enhanced stochastic gradient descent algorithm for robust training. However, the former mainly rely on a small set of labeled data, and the latter was unable to eliminate mislabeling. Hence, existing methods cannot meet the practicality of the authentication system.

(2) Insufficient coverage of human-device interaction patterns. The experiment in most studies [3–14] was conducted in a controlled laboratory environment with a small number of participants (7-50 people). Among most of the experiments [3–5, 7–11, 13, 14], all participants were asked to carry the mobile device and perform designated exercises (e.g., walking back and forth in a straight line at a constant speed) with prescribed time (ten minutes to several hours). This has the disadvantage that the coverage of human-device interaction patterns is insufficient since users may perform other behaviors such as sitting and running in daily life. Besides, previous studies [3–5, 8–12, 14] had a strong assumption that the device placement should be fixed. Moreover, the training set and test set in the above studies are performed under the same batch of users, and the performance of authentication models under real-world scenarios cannot be guaranteed (i.e., whether unknown/new attackers can bypass the authentication system). RISKCOG [2] and ESPIALCOG [1] experimented on a large dataset (1,513 people) with real-world noise for ten days to ensure the validity of the verification indicators. However, to solve the problem of an imbalanced dataset in binary classification models, the ratio of the number of samples from the user owner to that of other users was 1:5 in the above two studies [1, 2]. It will also lead to insufficient coverage of complex human-device interaction patterns in the real world.

(3) Insufficient transferability and privacy leakage. It is crucial to keep the transferability of the model and protect the privacy of user data in the mobile user authentication system. The idea of the previous work is mainly to generate a unique model for each user, it has the problem that when the service is commercialized and the number of users rises to hundreds of thousands or even millions, such computing costs are extremely huge. Moreover, in current implicit user authentication studies, all studies assume that users need to actively upload sensitive sensor data to the cloud to generate an authentication model, which will cause inconvenience to the user and the problem of privacy leakage.

(4) Low Accuracy. Accuracy, as well as True Positive Rate (TPR) and True Negative Rate (TNR), are important criteria for a comprehensive evaluation of the safety and usability of the authentication system. In most of the previous studies, the scarcity of both training and validation (i.e., data from unknown/new attackers) samples makes it difficult to fully demonstrate the practical ability in the complex environment. Though RISKCOG [2] and ESPIALCOG [1] obtained more credible data in the noisy environment, as well as a relatively higher accuracy (e.g., 81.41% TPR and 98.89% TNR for RISKCOG [2], 87.00% TPR and 97.93% TNR for ESPIALCOG [1]), they still fail to meet the high commercial safety standards.

Compared with the latest work ESPIALCOG [1] and RiskCog [2], our system has the following advantages: 1) We present a targeted optimized differential training method to eliminate the influence of man-made noisy labels. 2) We propose a siamese neural network with Long Short-Term Memory (LSTM) as sub-networks to deal with time-series data. It can maximize the coverage of available data, achieve sufficient coverage of interaction patterns, and realize the transferability of authentication models. 3) We deploy the well-trained model at the end-side to tremendously protect the privacy of users, reducing the time and economic cost of supplies to the greatest extent and improving the practicality of the system in real-world scenarios. All in all, we have made the following contributions.

- To eliminate the influence of man-made noise (noisy labels), we propose a targeted optimized differential training method based on the analysis of the data collection process in the real world. Differential training is fully automated in the process of label de-noising and does not require any domain knowledge or manual inspection. Moreover, it does not rely on any additional datasets with correct labels, which is better than methods [26, 27] relying on a small set of labeled data. Meanwhile, we improve the differential training to adapt to the problem of sample imbalance in binary classification in this system. It finally helps our system improve the accuracy by 1%.
- We design a siamese neural network structure based on LSTM for user authentication. By generating time-series data pairs, the coverage of training data is maximized (i.e., 100%), to achieve sufficient coverage of complex and diverse human-device interaction patterns. Besides, the network can learn contextual behaviors within the time-series data and distinguish the similarity of input data pairs. The above capabilities enable the corresponding model to have a strong transferability. Moreover, the locally deployed lightweight model tremendously protects the data privacy of the user, and greatly reduces the computing costs of suppliers.
- TRAPCOG can perform real-time mobile user authentication in a large scale real-world noisy dataset (1,513 people) with high accuracy. The experimental results show that the TPR, TNR, and ACC of TRAPCOG can reach 98.97%, 99.55%, and 99.26%, respectively. Fortunately, the accuracy will be further improved as the training set increases.

The rest of this article is organized as follows: In Section 2, we describe the TRAPCOG design in detail. Section 3 introduces the overall evaluation of our system. Section 4 discusses the strength and weaknesses of our work, and we also propose countermeasures. Section 5 surveys the related work and Section 6 summarizes our work.

2 SYSTEM DESIGN

In this section, we first introduce the threat model and overall architecture of our implicit user authentication mechanism for mobile devices, and then we discuss several key points in its design, including data collection, data de-noising, model construction, model training, and decision.

2.1 Threat Model and System Overview

Here, we give a living example to explain the threat model. Let us suppose Alice and Trudy are friends. One day, Alice forgot to turn off the screen of her smartphone when she left the classroom. At this time, Trudy could check the chat history of Instagram without the consent of Alice (i.e., Trudy was able to pick up Alice's smartphone and open the Instagram). In this case, the implicit user authentication system running in the background can detect the access of unauthorized users and then invoke the self-defense strategies, such as launching an alarm, rendering an invalid page, or requiring other authentication methods.

The architecture of TRAPCOG is shown in Figure 1. It contains three phases as follows:

In the collection phase: After the device supplier signing an agreement with a certain number of volunteers, the target application deployed on mobile devices will collect (the collection strategy will be introduced in Section 2.2) and upload time-series data from motion sensors (i.e., acceleration sensors, gyroscope sensors, and gravity sensors) to the supplier's cloud. Next, the data de-noising module will be invoked for eliminating the influence of machine abnormalities and man-made noises.

In the training phase: TRAPCOG will feed the uniformly formatted time-series data to the siamese neural network with LSTM as a sub-network for training. The trained model (with the ability of transferable) will be stored in the cloud to download for users who need authentication services.

In the authentication phase: Firstly, a new user should download the trained model (with an application) provided by the supplier to his/her smartphone and complete the initial data preservation stage (i.e., record his/her own human-device interaction pattern). After that, once the user opens applications (e.g., Instagram), the authentication process will be invoked. The application needs to detect the duration when the device is being actively used, because only the sensor readings during such a duration are effective to represent user's manner [2] (details in Section 2.2). The motion sensor data will be automatically collected, processed, and then passed into the trained model to compare with the user owner's sensor data. Finally, TRAPCOG will judge whether the current user is legitimate by the pre-defined threshold.

In reality, TRAPCOG can be used as a third-part service for implicit and accurate mobile user authentication in a real-time manner. Other explicit authentication methods or custom countermeasures can be leveraged if the authentication fails. We will describe the key components of TRAPCOG in the following article.

2.2 Data Collection and Formatting

For data collection, the readings of motion sensors (i.e., the acceleration sensor, the gyroscope sensor, and the gravity sensor) are selected as our data source due to their privacy insensitive (i.e., privacy-related permissions will not be invoked if the application wants to get the data from motion sensors). Therefore, the data format has 9 dimensions, as shown in Equation 1, where the subscripts a, gy, gr denote the acceleration, gyroscope and gravity. K denotes a mo-

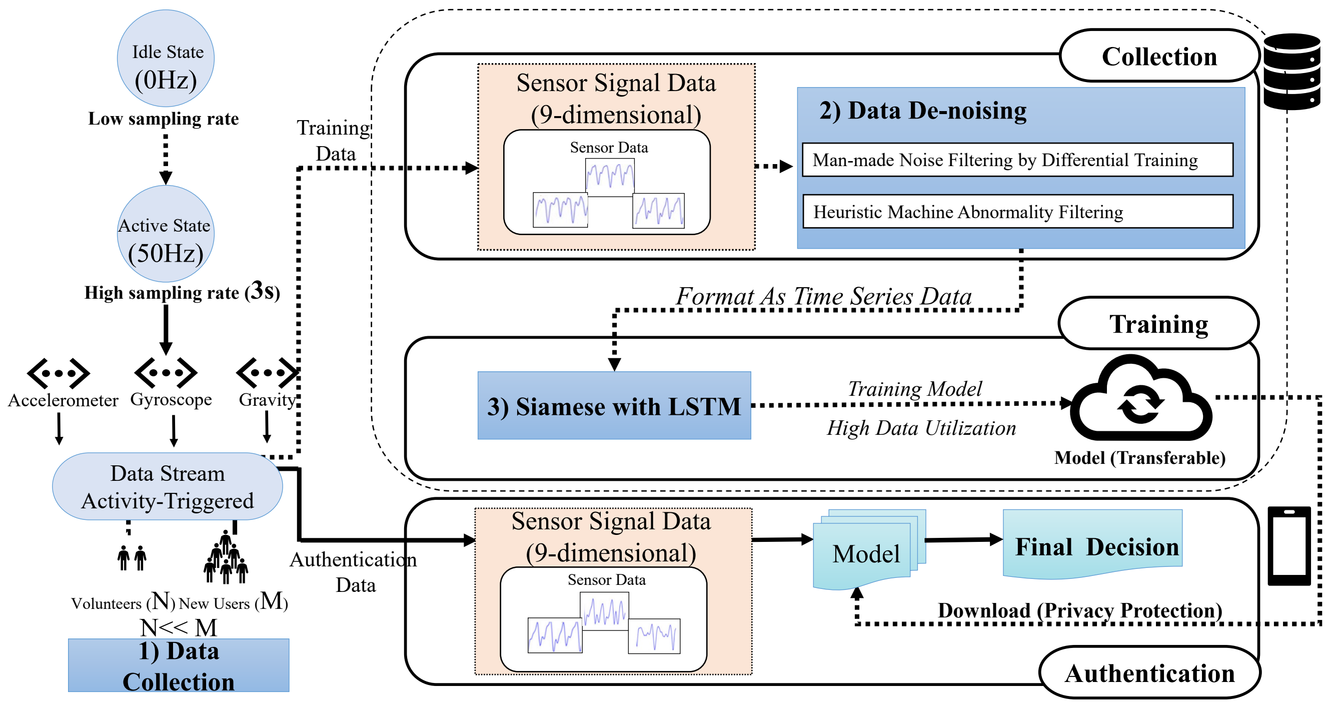


Fig. 1. System architecture: Including data collection, data de-noising, model training on the server side, model transfering, and real-time authentication on the user side. The information flow of volunteers during training phase is shown as a dashed line, and the information flow of new users during actual usage phase is shown as a solid line. The active state lasts for 3s with a frequency of 50 Hz to collect data from 3 sensors in 9-dimensional format.

ment, and X, Y, Z denote the three axes of the target sensor, respectively.

$$\begin{aligned}\{X_a(K), Y_a(K), Z_a(K),\\ X_{gy}(K), Y_{gy}(K), Z_{gy}(K),\\ X_{gr}(K), Y_{gr}(K), Z_{gr}(K)\}\end{aligned} \tag{1}$$

The collected data are from two places. As shown in Figure 1, one is volunteers' data for transferable model training, the other is new users' data for real-time authentication. TRAPCOG has two collection states: the idle state (the device screen is off or in a stationary state or no new application is running in the foreground) and the active state (the device screen is lightened and a new application is opened in the foreground). Specifically, the collection will only be enabled in the active state since only the sensor readings during such a duration are effective to represent the user's human-device interaction patterns. To reduce the battery consumption, we do not sample in the idle state and set the sampling frequency to 50 Hz in the active state (a new application is opened in the foreground and the device is in a non-stationary state). In addition, the duration of the data collection time is 3 seconds, which can best represent the user's human-computer interaction modality [2].

Finally, the data collected from each user are preprocessed and formatted into the uniform style, i.e., data of *3s (150 instances)* are evenly sliced into *3* equal-length data segments of *1.5s (75 instances)* with a 50% overlap. The final data segments of all users are two-dimensional vectors $[R, C]$, where R represents the sum of the selected sensor axes (9 in our system, i.e., 3 sensors, 3 axes per sensor), and C represents the size of the formatted data segments (75 instances in our system). Particularly, we use *data* to represent *data segments* in data de-noising and model construction/training. In addition, in the authentication phase, TRAPCOG will judge whether the 3 data segments after formatting are normal or not respectively, and determine the legitimate users with majority rule.

2.3 Date De-noising

For datasets collected in non-laboratory environments and applied to machine learning, the complex real-world scenarios introduce horrible noise. According to our observations, we can divide noisy data into two categories: *machine abnormality* and *man-made noise* (mislabeling). As we know, the noise in the dataset will significantly affect the performance of the authentication. Therefore, it is very important for us to adopt effective methods for the purpose of eliminating noise and improving the final accuracy. Next, we will describe how to eliminate them.

2.3.1 Machine Abnormality Filtering

This type of noise is caused by abnormal mobile device sensors. In hardware level, the built-in motion sensors of the mobile device will be affected by both internal factors (e.g., process, material, and design) and external factors (e.g., temperature, humidity, and service life). A certain deviation exists between the direct reading of the sensor and the true value, which is named machine abnormality. We identified three main types of anomalies: Equal-Value abnormalities, Jump-Point abnormalities and Zero-Value abnormalities.

Specific definitions and solutions can be found in previous studies [1, 2]. Note that this party is not the contribution of TRAPCOG, and we just reuse previous methods and delete these abnormalities in our experiment.

2.3.2 Man-made Noise Filtering

The man-made noise is caused by unrestricted human behavior during the data collection phase (i.e., data from volunteers in Figure 1). Since suppliers do not have any requirements for volunteers' daily behaviors when using the smartphone, it is necessary to train a robust model because the collected data will obviously contain various complex human-device interaction states. However, we consider the following scenario: Bob is a volunteer from the supplier and he participates in the data collection. One day, he lends his smartphone to his friend Charlie temporarily. During this period, the sensor data which is labeled as "Bob" is still uploading to the cloud. We call these part of data man-made noise. The noisy label will inevitably cause a mismatch in the training phase, thus affecting the final performance of the model.

We utilize differential training [28] and optimize it to reduce noisy labels. It is proven that the samples with true labels have better alignment and more similar distribution than the samples with random labels (or wrong labels), as shown in Section 3.3.2 for specific experimental results. Specifically, differential training will iteratively train two deep learning models, which can be any deep learning model of the same architecture. However, due to the binary imbalance of the data, differential training cannot be used directly, which requires to design and optimize solutions according to data characteristics. Therefore, we construct a label-balanced dataset by a heuristic solution to improve the effectiveness of differential training. For detailed solution, see the next paragraph (1) TD Construction. In our de-noising phase, we use binary classification to detect the noisy labels of each user. For each user, we treat the dataset of the user owner as Class 1 and that of other users as Class 0. To train the first model, the dataset contains the whole set of one selected user and data from some other users. While for the second one, we make use of a randomly down-sampled subset from the above dataset. For convenience, we call the dataset in the first model Total Data (TD), and the down-sampled subset as Subsample Data (SD). Similarly, the corresponding noise detection models are TD model, and SD model, respectively. Next, we will introduce how to use differential training for our data de-noising task.

(1) TD Construction. Recall that differential training will iteratively train two noise detection models. First, we should determine how to construct TD. In the iterative process of selecting users, let us suppose the currently selected user is K. The binary classification task will meet two problems as follows: 1) If we only take the data from user K for training, the convergence speed of the noise detection model will be too fast since all labels are 1, rendering the outlier detection algorithms fail to pick up the noisy data in a short loss vector (see step 3 Loss Vector Generation in (2) Detecting Noisy Labels of differential training for details.). 2) If we combine the data of K with all the data of other users as the model input, the imbalanced dataset will tremendously slow down the training speed. In order to solve the above

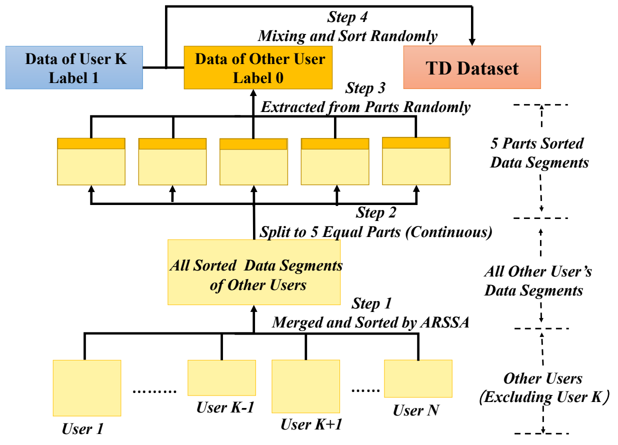


Fig. 2. Stratified sampling: including ARSSA sorting, data splitting, data extracting, and data mixing. In Step 1, we mix and sort all the users' data segments except User K. In Step 2, we divide the sorted data into 5 equal parts. In Step 3, we randomly select subsample from these 5 parts. In step 4, we mix and disorder the 5 subsamples. The goal of stratified sampling is to construct a positive-negative sample ratio of 1:1 dataset (TD) for user K.

problems, we use stratified sampling to meet the actual needs of differential training, as shown in Figure 2. We sort each data segment (from others) according to the average root sum square of acceleration (ARSSA) which has the highest fisher score [29]. We divide all sorted data segments into 5 equal parts (continuous), and randomly extract the same number of data segments from each part. Finally, we get the TD where the ratio of the number of positive samples to that of negative samples is 1:1.

(2) Detecting Noisy Labels. For each user/iteration, noisy labels will be detected and deleted. The iteration contains four steps including dataset down-sampling, training of TD and SD models, loss vector generation and outlier detection, as shown in Figure 3 and Algorithm 1.

- **Step 1 Dataset Down-sampling.** In step 1, the differential training randomly down-samples the TD of one user to form a smaller dataset SD. In our design, the ratio of the selected SD from TD is 0.2, satisfying the following requirement jointly [28]: 1) the size of SD needs to be as small as possible, and 2) SD will converge during the model training. Note that we will have 5 (1/0.2) SDs to cover all the data in TD.
- **Step 2 Training of TD and SD Models.** In step 2, we will train the two noise detection models "TD model" and "SD model" on the TD and SD, respectively. As mentioned before, these two models used in differential training can be any deep learning networks (e.g., Recurrent Nural Network, Convolutional Neural Network). In our system, considering the character of time-series data, we choose LSTM to detect noisy label. We refer the network structure in the well-known work in human activity recognition [30]. The number of layers (layer = 2) and the number of neurons in each layer are repeatedly used (Neuron = 32), with sigmoid layer as the output. We use the TensorFlow toolbox [31] to train two models, where all other parameters are set to their default values in the toolbox.

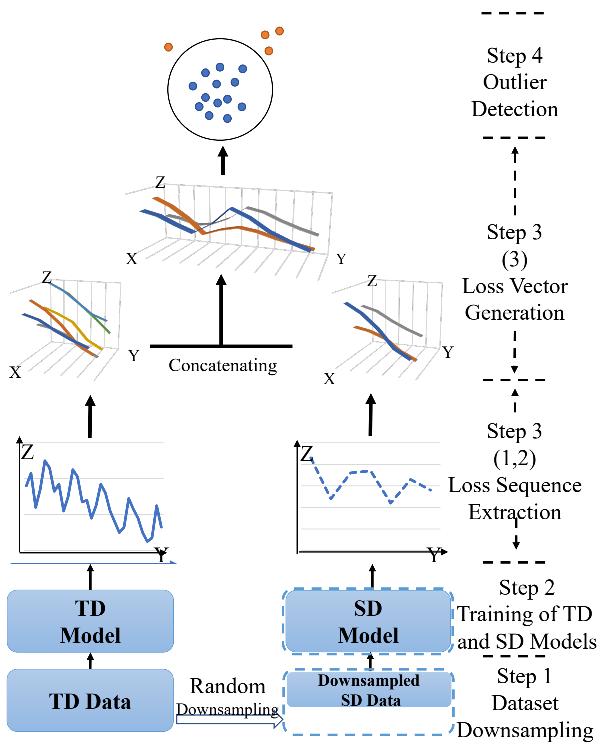


Fig. 3. Detecting noisy labels includes dataset down-sampling, TD and SD models training, loss vector generation and outlier detection. In the loss vector generation stage, X is for the index of data segment, Y is for the number of epochs, and Z is for the specific loss value. Note that the concatenating in Step 3 only operates on the data fragments with label 1 in the SD data.

- **Step 3 Loss Vector Generation.** In step 3, we aim to construct a loss vector for each data segment with label 1 in SD data. To this end, we 1) for each data segment in SD and TD, calculate the loss value for each epoch (the element in deep learning), 2) for each data segment in SD and TD, extract the loss values to form a loss sequence according to different epochs, 3) for each data segment with label 1 in SD, concatenate the loss sequence in SD with the corresponding one in TD to form a loss vector.
- **Step 4 Outlier Detection.** In step 4, a set of outlier detection algorithms are applied to the all loss vectors. For each data segment whose loss vector is detected as an outlier, its label is considered as ”wrong” and therefore it is likely to be deleted from the original dataset. Besides, we also delete noisy data segments with the probability equal to the dropout [32]. The dropout ratio is set to 0.5 in our system.
  Since most anomaly detection methods involve a containment rate as a threshold for identifying outliers.. In our system, inspired by the method proposed by Goldberger [33], the parameter containment of the outlier detection algorithm is set to $1-\alpha_{TD}$, where α_{TD} is the average accuracy from the 5-fold cross-validation on TD. In order to avoid any deviation of the single outlier detection algorithm, seven different outlier detection algorithms are used, and a majority vote is performed to obtain the final result of the outlier. With the help of the public toolkit named “PyOD” [34], Table 2 shows outlier detection algorithms used in our different training. Moreover, we set an early stopping mechanism in the training phase of noise detection models (i.e., the change of the loss value during the training phase is less than the predefined threshold, 0.0001 in our experiment), which greatly reduces the time required for differential training (the time of differential training in our system is reduced by 54%). In experiments, we enforce the stopping criterion through the API earlystop_callback() from the TensorFlow toolkit, where all parameters are set to their default values. Suppliers can adjust this threshold dynamically according to their demand.

Algorithm 1 Detecting Noisy Labels by Differential Training

```
Input:
All users' data are $U$, each user's data are $U_i$
Initialization:
Subsample_ratio.
Dropout_ratio.
List V.
1:  for $U_i \in U$ do
2:    construct dataset $TD_i$ by stratified sampling.
3:    construct loss sequence $L_t$ by training $TD_i$ model on dataset $TD_i$.
4:    for $j \in$ range $(1//subsample\_ratio)$ do
5:      $SD_j$ is $TD_i's$ $(j+1)th$ part.
6:      construct loss sequence $L_s$ by training $SD_j$ model on dataset $SD_j$.
7:      for $D_t \in SD_j$ do
8:        if $D_t$'s Label == 1 then
9:          construct loss vector $V_it$, corresponding to $D_t$ from $L_t$.
10:         construct loss vector $V_jt$, corresponding to $D_t$ from $L_s$.
11:         construct loss vector $V_t$ by concatenating $V_it$ and $V_jt$.
12:         put $V_t$ into $V$
13:       end if
14:     end for
15:     Get noisy data by applying outlier detection algorithms to V.
16:     Delete noisy data from $U_i$ with the probability equal to Dropout_ratio.
17:   end for
18: end for
```

TABLE 2
List of outlier detection algorithms used in differential training.

Angle-based Outlier Detector (ABOD)
Clustering Based Local Outlier Factor (CBLOF)
Histogram-based Outlier Detection (HBOS)
IsolationForest Outlier Detector (I-forest)
k-Nearest Neighbors Detector (kNN)
Local Outlier Factor (LOF)
Principal Component Analysis Outlier Detector (PCA)

2.4 Model Construction

Here, we construct a siamese neural network based on LSTM and we call it Siamese Neural Network with LSTM as Sub-Network. In this section, we will sequentially introduce the advantages and shortcomings of the LSTM network model, the siamese neural network with LSTM as a sub-network, and the construction of the training set and model training.

2.4.1 The Advantages and Shortcomings of Previous LSTM Model

LSTM is a kind of recurrent neural network, which is specially designed to solve the long-term dependency problem of general RNN. Meanwhile, binary classification has been widely used in previous user authentication studies [2, 13] since its ability for identifying accidental overlaps between different users. However, regarding combining binary classification with the LSTM network, as described in the state-of-the-art motion sensor based mobile user authentication study [1], we will meet the following two disadvantages:

Firstly, in order to ensure the efficiency of model training, and solve the problem of an imbalanced dataset, Zhu et al. [1] tried to construct the training set in which the ratio of the number of samples by user owner to that of other users is 1:5. Unfortunately, the data utilization rate is less than 0.4% (the total number of users is 1513). Obviously, such a low proportion of data utilization cannot fully cover the complex and changeable human-device interaction patterns in the real world. Secondly, when a new user wants to use the implicit user authentication service, he/she needs to upload a long period of time-series data to the server for model training with extra overhead. In addition, it also increases the leakage risk of the user's personal data.

2.4.2 Siamese Neural Network with LSTM as a Sub-network

Siamese neural network is based on the coupling framework established by two neural networks [35]. The Siamese neural network takes two samples as input and its two sub-networks each receives an input and outputs the corresponding representation embedded in the high-dimensional space. The distance between the two representations, such as Euclidean distance, is calculated to compare the similarity of the two samples.

Siamese neural network is very good at handling the issue where two inputs are "similar". A well trained siamese neural network will maximize the representation of different labels and minimize the representation of the same label [36]. Chorpa et al. [37] and Koch et al. [38] proposed siamese neural networks to learn a function that maps the input pattern to the target space so that the L1 norm in the target space approximates the one in the input space "semantic" distance, and they implemented one-shot learning on the Omniglot dataset [39] with the accuracy of about 92% which was as good as human beings.

However, these studies are only valid for image data and lack consideration of time-series data, which makes it hard to apply in our scenario. Therefore, we redesigned the architecture and chose LSTM as the sub-network of the siamese neural network, utilizing the ability of processing time-series data and distinguishing difference, to maximize the learning of behavior-sensitive time-series data. Since the user's behavior will be effectively represented by the time-series data, we can use this data to identify the legitimate user in real time.

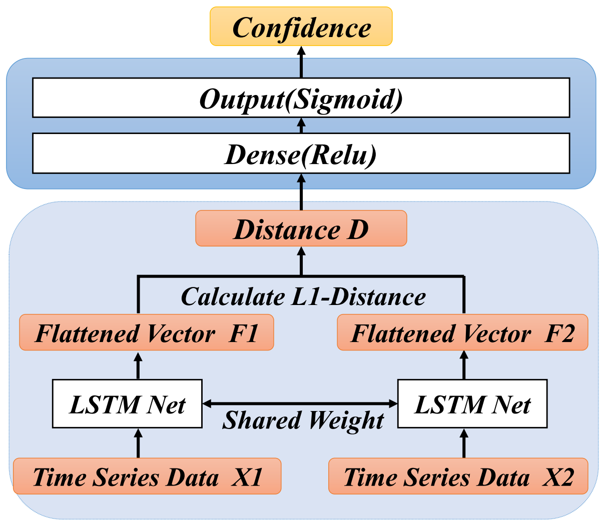


Fig. 4. The siamese neural network architecture with LSTM as sub-network.

Taking into account the specific application of our system, we propose a new model architecture, which is shown in Figure 4. Since the network structure is universal in similar tasks [30], the number of layers (layer=2) and the number of neurons per layer (neurons=32) are repeatedly used in our system. At the same time, in order to couple with the siamese neural network, the output layer of LSTM was fine-tuned. Besides, the original output layer (softmax) is changed to be a flattened layer, to output the stretched vector for later distance calculation.

2.5 Model Training

To train the model, we divide the dataset into two parts. On the total dataset of N people, we randomly select T users as the train set, and M users are used as the test set. The all data segments of each user have a uniform format so that can be input into the model. The final data segments of all users are two-dimensional vectors $[R, C]$, where R represents the sum of the selected sensor axes (9 in this system, i.e., 3 sensors, 3 axes per sensor), and C represents the size of the formatted data segments (75 in this system). For each iteration, we select three data segments of the user owner and one data segment of other users iteratively to feed the siamese neural network. Furthermore, four data segments are divided into two data pairs x_1 and x_2, the labels of x_1, and x_2 are 1, and 0 respectively. Here, label 1 means that the two data segments belong to the same user, and 0 is the opposite. In this way, we can 1) form a train set with a positive-negative sample ratio of 1:1, and 2) make full use of all datasets.

In the training phase, we input a pair of data segment x_1, x_2 into the network, the data x_1 enters the first siamese neural sub-network and output vector h_1, the data segment x_2 corresponds to the output vector h_2. Next, the

TABLE 3
The details of our datasets; all participants were skilled smartphone users with at least two years' experience; P for participants; NL for Noisy Label.

Dataset	# of P	Age	Provider	Phone Brands	Duration	NL
Dataset I	1513	20-60	Internet Company I	XIAOMI	10 days	Yes
Dataset II	8	20-30	Our Laboratory	HUAWEI	1 day	No
Dataset III	8	20-30	Our Laboratory	XIAOMI	1 day	No
Dataset IV	8	20-30	Our Laboratory	MEI	1 day	No
Dataset V	34	20-60	Internet Company II	Samsung	14 days	No

L1-distance D between two output vectors F_1 and F_2 is calculated by the distance calculation formula as follows: $d = \left(\sum_j \left| h_1^j - h_2^j \right| \right)$. Here $j = 0, 1, 2, ...J$, J is the length of the flattened vector. Finally, the L1-distance D is put into the fully connected layer with the activation function of RELU and the output layer with the activation function of Sigmoid to obtain the model confidence value $c = p\left(x_1^i, x_2^i\right)$, where function p represents in the i-th iteration model's judgment on the degree of similarity between the input x_1 and x_2, and c is a decimal between 0 and 1.

2.6 Decision

We set a threshold parameter t (t is from 0 to 1) to help authenticate. Users will be authenticated when c is greater than or equal to t. Otherwise, the user will be rejected. Note that the threshold t is adjustable. TRAPCOG can increase t if it is necessary to reject illegal users more strictly. Decreasing t means system is able to authenticate legal users with higher accuracy. We set $t = 0.5$ in our experiment.

3 EVALUATION

3.1 Dataset and Metrics

As shown in Table 3, we collected 5 datasets for our evaluation. Dataset I was directly collected from 1513 users through a large Internet company. Considering ethical risks, the user agreement of this collection included the purpose of this study. The sample frequency of motion sensors was 50HZ and the collection lasted for 10 days. Especially, IMEI was used to label each user. However, we did not use it directly but hashed it to protect the privacy of users. Besides Dataset I, we also collected data. i.e., Dataset II, Dataset III, Dataset IV, for mimic attacks in our laboratory (Section 3.5). Finally, Dataset V is also the pure date collected by Internet companyII to test de-noising ability of system. To eliminate random errors, we repeated all experiments three times.

To evaluate our model, we use the following metrics.

- **True Positive (TP):** The authorized owner is correctly identified.
- **False Positive (FP):** Other users are incorrectly identified as the authorized owner.
- **False Negative (FN):** The authorized owner is incorrectly identified as other users.
- **True Negative (TN):** Other users are correctly identified.
- **Effectiveness** The effectiveness contains True Positive Rate: $TPR = TP/\left(TP + FN\right)$, True Negative Rate: $TNR = TN/(TN+FP)$ and Accuracy: $Accuracy = (TP + TN)/(TP + FP + FN + TN)$.

TABLE 4
All parameters and hyper-parameters in this work.

Differential Training		Siamese With LSTM as Sub-network	
Deep Learning Model	LSTM	Deep Learning Model	LSTM
# of Layers	2	# of Layers	2
# of Neuron	32	# of Neuron	32
Optimizer	Adam	Optimizer	Adam
Learning Rate	0.0025	Learning Rate	0.0001
Batch Size	1	Batch Size	64
# of Stratified Sampling	5	Split of Training and Validation Set	8 : 2
# of Outlier Detection Algorithms	7	Threshold t	0.5
Subsample Ratio	0.2		
Dropout Ratio	0.5		

- **Overhead:** Time consuming of the training phase on the server and the authenticating phase on the user.

3.2 Hyper-parameters of Models

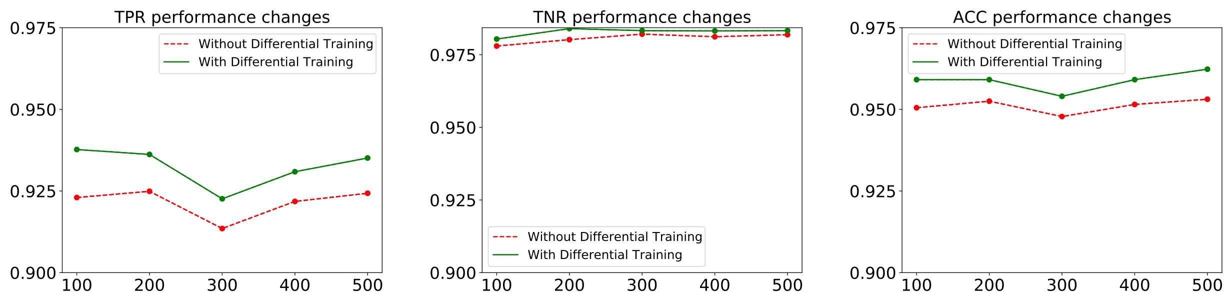
In this work, we first leverage the differential training to reduce the noisy data segments in the dataset. After that, with the help of a siamese neural network, we improve the coverage of human-device interaction patterns and build a transferable model. Both the differential training and the siamese neural network make use of LSTM as sub-networks. The hyper-parameters of these models are essential to achieve better effectiveness. The principle of hyper-parameter selection is to reuse existing/well-trained parameters, and then fine-tune these parameters by ourselves. Specifically, we follow previous research to initially set these hyper-parameters [1, 28, 38], and then use grid search to fine-tune the parameters. Due to space limitations, the tuning process of each parameter is not explained in detail here. Besides, we argue that hyper-parameter fine-tuning is not the most important part of our work because we would like to propose a generic implicit real-time authentication system rather than a set of specific hyper-parameters. To meet more requirements in the wild, suppliers who use TRAPCOG can fine-tune the hyper-parameters according to different demands. We list all hyper-parameters in TRAPCOG in Table 4.

3.3 Effectiveness

With the above-mentioned hyper-parameters, to evaluate the effectiveness of TRAPCOG, we will first demonstrate the effectiveness of differential training and then compare the overall effectiveness of TRAPCOG with other related work.

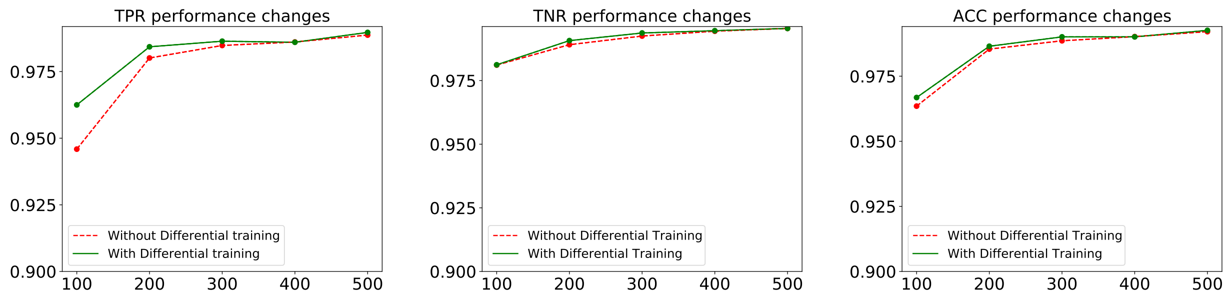
3.3.1 Construction of Training Set and Testing Set

We construct the training and testing sets on Dataset I following two facts that a) the amount of data with which a supplier can provide for training and the number of users to be authenticated in the wild are unbalanced and b) the latter will be much larger than the former. Therefore, in this work, we will train TRAPCOG with different users (T=100, 200, 300, 400, and 500) and test TRAPCOG on (M=100, 200, 300, 400, and 500) users, respectively. The size of the training set is 100 by default but we increase it gradually because we would like to help suppliers to choose the appropriate



(a) True Positive Rate (TPR) (b) True Negative Rate (TNR) (c) Accuracy (ACC)

Fig. 5. The changes of TPR, TNR, and ACC evaluated on the different sizes of testing sets by models trained from raw and denoised training sets (T = 100, M=100, 200, 300, 400, and 500).



(a) True Positive Rate (TPR) (b) True Negative Rate (TNR) (c) Accuracy (ACC)

Fig. 6. The changes of TPR, TNR, and ACC evaluated on the same testing set by models trained from different sizes of raw and denoised training sets (T = 100, 200, 300, 400, and 500, M = 500).

amount of users in the training. The size of the testing set is varied because we would like to simulate the process of user increment.

3.3.2 *Effect of Differential Training*

Experiment I: Trained on Dataset I, Tested on Dataset I. To evaluate the effect of the differential training, we first train TRAPCOG on the raw training set and the de-noised training set (T=100), respectively. We evaluate two models on the same testing set (M=100, 200, 300, 400, and 500). As shown in Figure 5, no matter what the size of the testing size is, the TP, TN, and ACC of the denoised model are all higher than the model trained with raw data. Second, we train TRAPCOG on the raw and de-noised training set with different users (T=100, 200, 300, 400, and 500) and evaluate two models on the same testing set (M=100). The results are shown in Figure 6. As the size of the training set increases, the TPR, TNR, and ACC of the model are gradually increasing. Mostly, with a smaller number of users, the accuracy improvement brought by the differential training is greater. The possible reason is that the less training data, the greater the impact of mislabeling on the model.Besides, as the evaluation index approaches the maximum value of 100%, marginal effects exist.

We believe that if the size of the testing set is much larger than the training set, the differential training will bring greater improvement. Additionally, differential training has a more obvious effect on the improvement of TPR, showing its boost to the identification of legitimate users and the low TP in the previous work [1]. This is because the differential training is to help eliminate the data segments that do not belong to a user. Based on the two experiences, we can conclude that differential training can effectively solve the mislabeling problem in the dataset and improve the effectiveness of models.

Experiment II: Trained on Dataset V, Tested on Dataset I. Besides, we conduct the experiment by adding noisy labels to the pure dataset to demonstrate that differential training outperforms previous related work [1, 2, 30]. In our evaluation, we implement man-made noise addition (i.e., 0% noisy data, 15% noisy data, and 30% noisy data) by randomly adding the data of other users to the pure data of target user, and compare the above datasets using four different methods (including differential training), respectively.The results of total test accuracy are shown in Table 5. It should be noted that the model in this experiment is trained on Dataset V (only 34 persons) and evaluated on the test set (M=500).

From Table 5, we can see that differential training has stronger de-noising ability than other related works. As the ratio of noisy labels increases, the performance of differential training is even better, which can even improve the accuracy by nearly 4%. This result also confirms our analysis above. Furthermore, differential training does not require an additional clean dataset compared to the other two methods [1, 2], which helps to reduce the overhead and

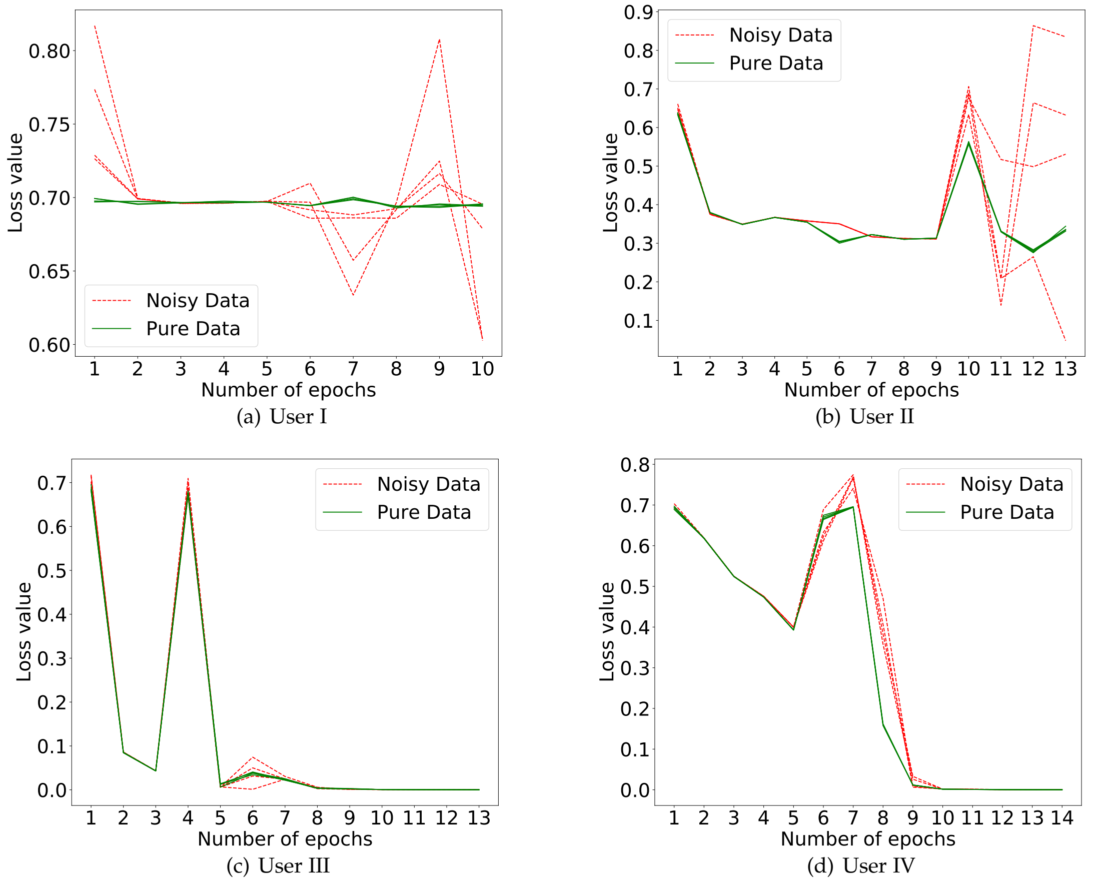


Fig. 7. The figure of loss vectors generated in the differential training. Different subplots represent different users, and there are 8 loss vectors in each subplot. The horizontal axis indicates the total number of epochs fitted by the user's data segments in the training TD and SD model, and the vertical axis indicates the loss value of a data segment during training. The red dashed line indicates the man-made noisy data, and the green solid line indicates the pure data.

TABLE 5
The Overall Accuracy of Differential Training Compared With Other Related Work on Dataset V. Where noise addition is adding man-made noise, i.e. adding other users' data to the pure data of one user.

Studies	**Dataset V**			**Pure Data Needed**
	Original	15% noise	30% noise	
Baseline [30]	94.76%	90.11%	85.79%	/
Semi-supervised Learning [2]	94.72%	92.27%	87.20%	Yes
Enhanced SGD Algorithm [1]	**94.92%**	91.62%	87.73%	Yes
Differential Training	94.83%	**93.07%**	**90.49%**	**No**

TABLE 6
The specific degree of the differential training to deal with the mislabeling phenomenon of the training set.

# of Users	**# of Total Data Fragments**	**Avg of Data Fragments**	**# of Noisy Labels**	**% of Noisy Labels**
100	93,728	937	4,708	5.02%
200	185,274	926	8,238	4.45%
300	278,302	927	13,064	4.69%
400	378,864	947	17,175	4.53%
500	482,889	965	21,429	4.44%

remove the reliance on additional datasets that may lead bias.

Experiment III: Loss vector visualization. To visualize the differences of the loss vectors obtained in the differential training, we conducted experiments strictly following the hyper-parameters in Section 3.2 and performed differential training on four randomly selected users. We finally obtained the loss vectors corresponding to all data segments (353, 1991, 969, and 1502, respectively) one by one under different users. It should be noted that the number of epochs required to fit the TD model and SD model varies among users, and therefore the dimension of vectors varies (10, 13, 13, and 14, respectively). We selected 8 loss vectors with balanced positive and negative labels from each user, and the results are shown in Figure 7. We can see that there is a difference in the distribution of the loss vectors corresponding to noisy and pure data. In addition, the distribution of the loss vectors corresponding to the pure data is almost same. Therefore, we can use the outlier detection algorithms (Step 4 in Section 2.3.2) to identify and filter out the noisy data segments for each user. It should be noted that a loss vector is a concatenation of the loss values corresponding one data segment in the training phase of TD and SD models, so there will be a large jump value (e.g. epoch=4 in Figure 7(c)).

Finally, to detail the changes of the training set by the

TABLE 7
The performance of TRAPCOG compared with previous related work; The coverage refers to the proportion of Dataset I used for training one model; The effectiveness of transferability refers to the performance of the system on a dataset consisting of non-trainers; The data of users needed refers to whether the system requires users to provide their own data to retrain the model or extract features for optimal performance.

Studies	Accuracy	Coverage	Offline Verification Letency	Effectiveness of Transferability	Data of users Needed
TRAPCOG	TNR=99.55% TPR=98.97%	100%	3474ms	TNR=99.55% TPR=98.97%	No
Lee et al.(2017)	73-96%	0.06%	4000ms	TNR=7.6-92.6% TPR=0%	Yes
Buriro et al.(2017)	TPR=96% EER=4%	0.06%	NA	TNR=4-96% TPR=4%	Yes
Shen et al.(2018)	TNR=94.97% TPR=96.02%	0.06%	8000ms	TNR=5.03-94.97%% TPR=3.98%	Yes
Zhu et al. (2019)	TNR=98.89% TPR=81.41%	0.39%	3237.7ms	TNR=1.01-98.89% TPR=18.59%	Yes
Zhu et al. (2020)	TNR=97.93% TPR=87.00%	0.39%	3124.6ms	TNR=2.07-97.93% TPR=13.00%	Yes

differential training, we list several numbers in Table 6. The specific content of the table is the number of specific users in the training set, the number of data segments, the average number of each user's data segments, the number of noisy labels, and their proportion. In general, differential training can reduce around 5% of noisy data segments upon our dataset. Note that the specific number here is also calculated by repeating this experiment three times.

3.3.3 *Overall Effectiveness of* TRAPCOG

We also compare the effectiveness of TRAPCOG with previous related work [1, 2, 12–14].

- **Best Performance.** We train TRAPCOG with 500 random users and test it with 500 different users. Then we use the obtained evaluation result as the best performance indicator of this system. As for other studies, due to the lack of datasets, we only list their evaluation metrics, coverage and verification latency here. The overall results are shown in Table 7. We can see that both the TPR and TNR of TRAPCOG are much higher than the previous ones. In addition, since TRAPCOG has more parameters, the offline verification latency is slightly longer than the latest related work [1, 2]. However, the gap is only within 350ms, an increase of less than 10%, which is completely acceptable for users. In detail, TRAPCOG combines the ability of LSTM to learn data and the ability of the siamese neural network to judge the similarity and difference of the data, which achieves high accuracy.
- **Transferability.** Transferability means that the pre-trained siamese neural network can be directly shared and used by all users, without the need for each user to train the model separately. As shown in Table 7, the Effectiveness of Transferability of TrapCog is much higher than existing work [1, 2, 12-14]. Consider above existing studies, we find that if the model trained by user A is used directly for other users (e.g., user B), the TPR will be extremely low and the TNR will fluctuate greatly (it is not difficult to see that when user B authenticates user A with user A's model, TNR is the lowest). The main reason is that previous studies need to build a targeted authentication model for each user, and their models lack transferability. In addition, in order to achieve a better authentication performance, existing studies always require users to upload their own sensor-sensitive data and re-extract features, which increases the complexity of usage, the risk of data exfiltration, and the overhead of service provider. On the contrary, our system eliminates the need for repeated training of the model for new users in traditional methods by deploying a transferable model that learns the ability to distinguish between dual inputs. In short, our system does not require users to upload sensitive data to the cloud, which eliminates the problems of user inconvenience and privacy leakage. To the best of our knowledge, this is the first work to achieve robust transferability in dynamic biometric-based mobile user authentication.

3.4 Usability Analysis

In this section we will use third-party metrics to evaluate the usability of TRAPCOG. Third-party metrics can be divided into theoretical derivations [40–43] and questionnaires [44, 45], the latter being more suitable for evaluating the usability of TRAPCOG. Therefore, we use Sample Acquisition Time and System Usability Scale (SUS) [46] to perform usability analysis.The former represents the system latency on necessary data sampling, and the latter represents the user's quantitative assessment of system usability.

We invited 34 users from dataset V to take the SUS questionnaire after using TrapCog for two weeks. It contains 10 items (e.g., whether the user would like to use it often, whether the user feels bothered) and users have five response options ranging from strongly agree (score 10) to strongly disagree (score 0). Finally, we obtained the performance of different authentication systems [1, 2, 47] on sample acquisition time and SUS score as shown in Table 8. We can see that in terms of sampling time, TrapCog belongs to the first tier (3.5s), only 0.4 seconds behind the fastest EspialCog (3.1s), which is perfectly acceptable. Moreover, TrapCog is the only one to score over 90 on the SUS score (91), outperforming existing mobile user authentication systems. After analyzing the results, we concluded that users do not need to wait long time for the model training compared to EspialCog [1] and RiskCog [2], and users do not need to interact explicitly compared to Password, Voice and Face [47], etc. Most users appreciate the usability and effectiveness of TrapCog.

3.5 Security Analysis

In this section, we demonstrate the security of TRAPCOG in terms of both mimic attacks and information entropy.

TABLE 8
Sample acquisition time, SUS score for different authentication systems. The range of SUS score is from 0 to 100.

Method	Sample AcquisitionTime (s)	SUS Score
TrapCog	3.5	91
RiskCog	3.2	87
EspialCog	3.1	84.5
Password	4.3	78
Voice	5.4	66
Face	3.9	75
Gesture	4.2	77
Face + Voice	5.3	46
Gesture + Voice	4.7	50

3.5.1 *Resistance to Two Mimic Attacks*

To attack implicit real-time authentication systems, a malicious user would like to choose passive zero effort imposter attacks and active imposter mimic attacks [48]. The former is common in the wild because anyone can pick up one's smartphone and try to use it. To fight against this attack, the well-trained model requires the ability to distinguish other users from the owner of the smartphone. The effectiveness of TRAPCOG described in Section 3.3 has demonstrated this ability. Therefore, TRAPCOG can handle this attack. The latter (mimic attacks) is more sophisticated and we will analyze it in the following.

To simulate mimic attack in the wild, we first randomly select one person as the victim and then ask remaining 7 person to observe victim's human-device interaction patterns for one day. Next, each person (one of remaining 7 person) tries to mimic the patterns of the victim for 60 times (3 seconds each time). The dataset II, III, and IV in Table 3 gives the details about our mimic experiment. Note that 8 volunteers in this experiment did not participate in the previous model training process, since TRAPCOG is transferable, we only need to test the well-trained model mentioned in Section 3.3 with the newly collected datasets and then determine whether the model can correctly judge illegal users in the datasets. To improve the reliability, we repeated the above experiments on three different brands of mobile phones. As shown in Table 12, TRAPCOG prevents mimic attacks with the probability of more than 99%. Therefore, it is difficult to bypass our authentication via mimic attacks.

3.5.2 *Security from Information Entropy Perspective*

In this section, we will compute the information entropy of TRAPCOG and compare it with other mobile authentication methods to demonstrate the security of our system from the Information entropy perspective.

According to Shannon's definition [49], information entropy is a measurement of information uncertainty in a space. We demonstrate the effectiveness of TRAPCOG on authentication and resistance to attacks by calculating the information entropy of the input space. As described in Section 2.2, we finally obtain data segments in the format *[R, C]*, which are normalized to 10 decimal places of accuracy. In addition, as described in Section 2.3.1, there are no jump point abnormality in the data, so we consider the data to be a high-dimensional continuous random variable. We decide to estimate the information entropy by using the classical

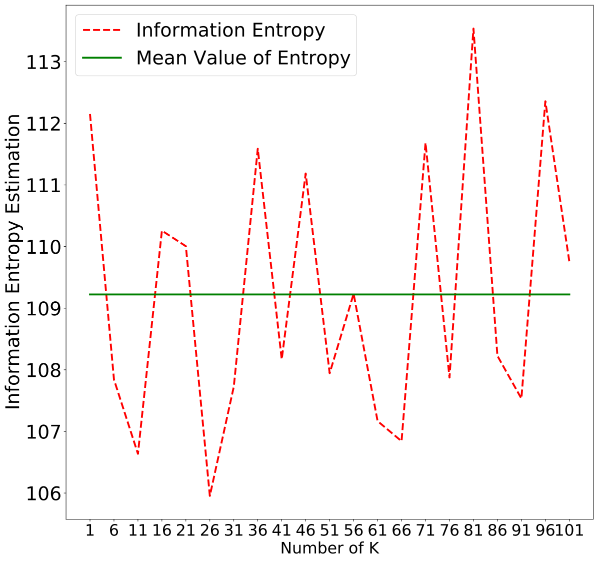


Fig. 8. The comparison figure of information entropy for different k values. The horizontal coordinate indicates the different k values, the vertical coordinate indicates the estimated information entropy, the red dashed line indicates the change of information entropy, and the green solid line indicates the mean value of information entropy.

KNN-Nearest Neighbor Estimator [50], which has been widely used in various research problems.

Assume there are N i.i.d. D-dimensional samples $x_1, ..., x_N \sim P$, where the probability density function p is unknown. The classical KNN estimator is written as:

$$H(X) \approx \psi(N) - \psi(K) + log(c_D) + \frac{D}{N}\sum_{i=1}^{N} log(\varepsilon_i) \quad (2)$$

where ψ denotes the digamma function, Γ denotes the gamma function, ε_i denotes the Euclidean distance from x_i, to its nearest neighbor, $c_D = \frac{\pi^{\frac{D}{2}}}{\Gamma(1+\frac{D}{2})}$.

We estimate information entropy on *480,000* data segments based on *500* users in Table 6 and perform a comparison experiment on parameter K (1-101) as shown in Figure 8. We can see that the final mean value of information entropy is obtained as *109.22*, which is much higher than other mobile authentication methods as shown in Table 9. Therefore, we believe that TRAPCOG is sufficiently secure from the perspective of information entropy.

3.6 Overhead

The deployment model of TRAPCOG is introduced in Section 2.1. Here, we will present the overhead of the training phase on the server and the overhead of the authentication phase on the user.

3.6.1 *Overhead on the Server*

We train our model (TensorFlow) on a server equipped with Intel(R) Core(TM) i9-10900X CPU and 128G memory running on Ubuntu 18.04. Table 10 lists the average

TABLE 9
The comparison table of entropy calculated under different mobile authentication methods.

Study	Mobile Authentication Method	Entropy (bits)
TrapCog	Gesture Authentication	**109.22**
Wang et al. [51]	Chosen 4-digit PINs	8.41
	Chosen 6-digit PINs	13.21
Wang et al. [52]	Chosen Passwords	20-22
Inthavisas et al. [53]	Voice Authentication	18-30
Feng et al. [54]	Facial Authentication	75
Sadeghi et. al. [55]	EEG Authentication	83

TABLE 10
The server overhead of the previous work and TRAPCOG under different size of training sets; T means that the training set consists of T users.

Studies	T	Training Once	Iterations	Training Util Convergence
ESPIALCOG [1]	6	3.13s	1500	1.3h
TRAPCOG	100	2,706.83s	23	17.3h
TRAPCOG	200	6,933.89s	19	36.6h
TRAPCOG	300	10,499.21s	19	55.4h
TRAPCOG	400	13,379.32s	22	81.8h
TRAPCOG	500	18,011.41s	30	150.1h

training overhead under different numbers of users with GPU acceleration and comparison with the previous work ESPIALCOG [1] (with GPU acceleration also). Using GPU acclamation is reasonable because the model training is very time-consuming in this work. Nowadays, the GPU acceleration service is common and without GPU acceleration, the time-consuming task will be unacceptable. As shown in Table 10, we can see that the one-time training of TRAPCOG is much longer than that of the previous. That is because TRAPCOG has a very high utilization rate of data, and the amount of data that needs to be trained far exceeds the previous work. Moreover, the model's complexity of TRAPCOG is higher than that of the previous work. When it comes to deploying TRAPCOG in the wild, the model can be trained for only once since the transferability of TRAPCOG, the supplier will be tolerant of the time-consuming training. In the case of serial computing, as long as a few hundred users require ESPIALCOG to train their models, the time will exceed that of TRAPCOG. Even if the distributed computing technology is used, once the number of users exceeds tens of thousands, the training time of ESPIALCOG will far exceed as well.

3.6.2 Overhead on the Client

On the user (Android smartphones), we also deploy TensorFlow [1] for the authentication phase. As shown in Table 11, we monitored battery consumption, CPU and memory usage during the idle state, the data collection state, and the offline authentication state, respectively. The data indicate that the overhead of the offline authentication state is the highest, but this phase only lasts 3 seconds. Besides, it should be noted that the power consumption is collected by the participants using their mobile phones normally, because our authentication system is not continuous but heuristic (details in Section 2.2). Moreover, the battery required by our system in an hour is less than 1.5 percent. Finally, the size of the uncompressed model in our system is only 1.09MB , which is light enough to download and install in smartphone. In short, these overheads have no effect on the normal use of smartphones.

1. Tensorflow. https://www.tensorflow.org/guide/saved_model

4 DISCUSSION

In this section, we will discuss the advantages and disadvantages of TRAPCOG.

4.1 The Advantages of TRAPCOG

High coverage of human-device interaction patterns, strong transferability and privacy-preserving. We design a LSTM based siamese neural network that can not only make full use of all available data segments, but also expand the trainable data by generating time-series data pairs. Moreover, it can reduce the cost of collecting training data, and maximize the coverage of human-device interaction patterns. Besides, the network has the ability to learn contextual behaviors within the time-series data and distinguish the similarity of input data pairs. This ability can eliminate the time cost for uploading data to the cloud, protect the privacy of user's information, and significantly reduce the computing cost of the supplier.

Strong de-noising ability. We use differential training to reduce noisy labels. In the real-world scenario, the specific behaviors of users during the data collection period cannot be determined, which will undoubtedly affect the accuracy of the final model. To solve this problem, we utilize differential training, which is fully automated in the process of label de-noising and does not require domain knowledge or manual inspection. In addition, it does not rely on any additional datasets with correct labels.

High accuracy. TRAPCOG can perform real-time mobile user authentication in a large scale real-world noisy dataset with high accuracy. The evaluation results indicator that the TPR and TNR of our system are much higher than the previous studies. In particular, the significant lead of TPR indicates that our system has solved the usability problem (false negative) caused by previous work.

4.2 The Weaknesses of TRAPCOG and Future Work

Firstly, although new users do not need to upload data to the cloud, users still need to provide their own data in advance for subsequent comparison in the mobile side. In this article, we do not describe in detail how to ensure that the data stored in the mobile device are not illegally tampered with during the initial use stage. In fact, we can collect data in the initial phase with the help of other authentication methods, and store the collected data into a secure area of the smartphone, such as TrustZone.

Secondly, no detailed experiment is carried out for the problem that the change of the human-device interaction mode of the same user at different times may lead to the inability to correctly authenticate legitimate users. In other words, further experiments are needed to explore the performance of the model over a longer time span (i.e., one month or more).

TABLE 11
The overhead results on four different smartphones; the measurements of CPU and memory are averaged over 1000 times of authentications.; the measurement of battery consumption lasts for five hours.

Phone Brands	**Idle State**		**Data Collection**		**Offline Authentication**		**Battery Consumption (mAh)**
	CPU(%)	Memory(MB)	CPU(%)	Memory(MB)	CPU(%)	Memory(MB)	
IQOO Neo5	0.4	1.3	1.9	15.5	13.0	88.3	304.9/4400
Mate 40	0.2	0.7	1.7	14.1	10.9	82.4	295.6/4200
MI 10	0.6	1.9	2.6	16.8	14.1	90.6	337.8/4780
Samsung S21	0.5	1.8	2.1	16.3	12.7	79.9	287.2/4000

TABLE 12
TRAPCOG's resistance to mimic attacks under different smartphone brands.

Dataset	**Phone Brands**	**TNR**
Dataset II	HUAWEI	99.78%
Dataset III	XIAOMI	99.84%
Dataset IV	MEI	99.16%

Thirdly, TRAPCOG can not accurately authenticate users on mobile devices in stationary state where sensor data remains unchanged. In such circumstances, we suggest resorting to traditional authentication methods (e.g., fingerprint authentication and face authentication) as a supplementary measure. Furthermore, we propose that incorporating data from finger swipes or screen pressure can surmount the limitations of TRAPCOG in this usage scenario.

5 RELATED WORK

In this section, we will introduce related work in implicit authentication and compare it with TRAPCOG. The specific comparison aspects include human-device interaction pattern coverage, data de-noising ability, and model transferability, which reflects the advantages of our work.

- **Improve the coverage of human-device interaction patterns.** The traditional data collection method is to invite participants to provide sensor data in certain specific situations, such as walking [3–5, 7, 9, 10], going up/down stairs [4], picking up a mobile device [6, 12] and touching the screen of the mobile device [11, 13, 14]. Both the number and complexity of the human-device interaction modes in the real world far exceed the data collected from a laboratory. Therefore, the model obtained with these limited data will not meet the practicality of implicit real-time authentication in the wild. In the work of Zhu et al. [1, 2], a large dataset of thousands of people was collected. However, its model could just use the data of only 6 persons in the dataset and was limited with the imbalance problem of binary classifications. Therefore, although the user behavior is no longer restricted when collecting data, the model's narrow coverage of patterns and low data utilization is still the barriers before the deployment of such systems in the wild. The LSTM-based siamese neural networks proposed in this work can solve this problem and use all the collected data. That is to say, it can cover the complex and changeable human-device interaction patterns making TRAPCOG feasible to be deployed in the wild.
- **Solve the reduction of man-made noises.** Most previous researches on user authentication based on motion sensors has rarely considered the influence of the noisy data [3–14]. In other words, they all believe that the collected data is pure and representative of the user. However, the data collected by mobile devices in the real world will have various noises. Zhu et al. [1] observed data abnormalities caused by various hardware problems in the real dataset, and divided them into equal-value abnormalities, jump-point abnormalities, and zero-value abnormalities. In addition, for man-made mislabeling problems, they adopted a semi-supervised learning method and an enhanced SGD algorithm. However the former required a pure dataset and the latter performed poorly. We instead use differential training to reduce the noisy data and achieved a promising improvement in model evaluation.
- **Realize the transferability of the model and the privacy protection of users.** Within the scope of our investigation, although this is a real problem, none of the work mentioned this aspect [3–14, 56–61]. At present, almost all implicit real-time user authentication systems train one model for one user. The trained model has good TPR and TNR only for specific user. Therefore, suppliers need to separately train and provide customized models for each user. The biggest problem with this approach is that the high cost and leakage of privacy. When the implicit user authentication service is open to the public, it is necessary to train and save the model separately for hundreds of thousands of users. At this time, the cloud computing cost and the hardware cost required to save the model are extremely expensive. In addition, the user needs to provide its data to train a model for others, which causes privacy leakage. Our system is different from previous work that focuses on classifying the owner to any others. TRAPCOG will first collect training data from volunteers and train a transferable model. Then, suppliers can push this model to users. The data of each user will be saved locally because it will not be used in the model of other users. Our model is transferable and can protect the privacy of users.

6 CONCLUSION

In this paper, we propose TRAPCOG that uses a siamese neural network based on LSTM to perform implicit real-time mobile user authentication. TRAPCOG is different from previous studies where data de-noising is not enough, human-device interaction patterns coverage is insufficient, models are not transferable, and user privacy is not well protected. We solve the problem of mislabeling by differential training based on down-sampling and make full use of the dataset to maximize the coverage of the real-world human-device interaction patterns by a siamese neural network with LSTM as sub-networks. At the same time, since the model has the ability to distinguish the degree of similarity of the input, we realize the transferability of the model. Under this circumstance, users only need to deploy our system and authenticate locally on the user. This architecture can protect the privacy of users because no private data should be uploaded. In addition, TRAPCOG has extremely high effectiveness and accuracy. The experimental results on a large and noisy dataset show that our system obtains 98.97% TPR, 99.55% TNR, and 99.26% authentication accuracy, far exceeding existing studies. The results also show that the accuracy is still rising with the increase of the training set. Finally, TRAPCOG also has excellent resistance to mimic attacks with a recognition rate of over 99%. Therefore TRAPCOG outperforms latest studies and meets the requirements of security, privacy, and usability in mobile user authentication, which is feasible to deploy in the wild.

REFERENCES

[1] T. Zhu, Z. Weng, Q. Song, Y. Chen, Q. Liu, Y. Chen, M. Lv, and T. Chen, "Espialcog: General, efficient and robust mobile user implicit authentication in noisy environment," *IEEE Transactions on Mobile Computing*, vol. 21, no. 2, pp. 555–572, 2020.

[2] T. Zhu, Z. Qu, H. Xu, J. Zhang, Z. Shao, Y. Chen, S. Prabhakar, and J. Yang, "Riskcog: Unobtrusive real-time user authentication on mobile devices in the wild," *IEEE Transactions on Mobile Computing*, vol. 19, no. 2, pp. 466–483, 2019.

[3] M. O. Derawi, C. Nickel, P. Bours, and C. Busch, "Unobtrusive user-authentication on mobile phones using biometric gait recognition," in *2010 Sixth International Conference on Intelligent Information Hiding and Multimedia Signal Processing*. IEEE, 2010, pp. 306–311.

[4] J. R. Kwapisz, G. M. Weiss, and S. A. Moore, "Cell phone-based biometric identification," in *2010 Fourth IEEE International Conference on Biometrics: Theory, Applications and Systems (BTAS)*. IEEE, 2010, pp. 1–7.

[5] C. C. Ho, C. Eswaran, K.-W. Ng, and J.-Y. Leow, "An unobtrusive android person verification using accelerometer based gait," in *Proceedings of the 10th International Conference on Advances in Mobile Computing & Multimedia*, 2012, pp. 271–274.

[6] J. Zhu, P. Wu, X. Wang, and J. Zhang, "Sensec: Mobile security through passive sensing," in *2013 International Conference on Computing, Networking and Communications (ICNC)*. IEEE, 2013, pp. 1128–1133.

[7] H. Lu, J. Huang, T. Saha, and L. Nachman, "Unobtrusive gait verification for mobile phones," in *Proceedings of the 2014 ACM international symposium on wearable computers*, 2014, pp. 91–98.

[8] H. G. Kayacik, M. Just, L. Baillie, D. Aspinall, and N. Micallef, "Data driven authentication: On the effectiveness of user behaviour modelling with mobile device sensors," *arXiv preprint arXiv:1410.7743*, 2014.

[9] Y. Ren, Y. Chen, M. C. Chuah, and J. Yang, "User verification leveraging gait recognition for smartphone enabled mobile healthcare systems," *IEEE Transactions on Mobile Computing*, vol. 14, no. 9, pp. 1961–1974, 2014.

[10] W.-H. Lee and R. B. Lee, "Multi-sensor authentication to improve smartphone security," in *2015 International conference on information systems security and privacy (ICISSP)*. IEEE, 2015, pp. 1–11.

[11] Z. Sitová, J. Šeděnka, Q. Yang, G. Peng, G. Zhou, P. Gasti, and K. S. Balagani, "Hmog: New behavioral biometric features for continuous authentication of smartphone users," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 5, pp. 877–892, 2015.

[12] W.-H. Lee, X. Liu, Y. Shen, H. Jin, and R. B. Lee, "Secure pick up: Implicit authentication when you start using the smartphone," in *Proceedings of the 22nd ACM on Symposium on Access Control Models and Technologies*, 2017, pp. 67–78.

[13] A. Buriro, B. Crispo, and Y. Zhauniarovich, "Please hold on: Unobtrusive user authentication using smartphone's built-in sensors," in *2017 IEEE International Conference on Identity, Security and Behavior Analysis (ISBA)*. IEEE, 2017, pp. 1–8.

[14] C. Shen, Y. Li, Y. Chen, X. Guan, and R. A. Maxion, "Performance analysis of multi-motion sensor behavior for active smartphone authentication," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 1, pp. 48–62, 2017.

[15] Z. Zhao, G.-J. Ahn, and H. Hu, "Picture gesture authentication: Empirical analysis, automated attacks, and scheme evaluation," *ACM Transactions on Information and System Security (TISSEC)*, vol. 17, no. 4, pp. 1–37, 2015.

[16] D. Nyang, A. Mohaisen, and J. Kang, "Keylogging-resistant visual authentication protocols," *IEEE Transactions on Mobile Computing*, vol. 13, no. 11, pp. 2566–2579, 2014.

[17] N. H. Zakaria, D. Griffiths, S. Brostoff, and J. Yan, "Shoulder surfing defence for recall-based graphical passwords," in *Proceedings of the seventh symposium on usable privacy and security*, 2011, pp. 1–12.

[18] D. Nyang, H. Kim, W. Lee, S.-b. Kang, G. Cho, M.-K. Lee, and A. Mohaisen, "Two-thumbs-up: Physical protection for pin entry secure against recording attacks," *computers & security*, vol. 78, pp. 1–15, 2018.

[19] A. De Luca, A. Hang, F. Brudy, C. Lindner, and H. Hussmann, "Touch me once and i know it's you! implicit authentication based on touch screen patterns," in *proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2012, pp. 987–996.

[20] N. L. Clarke and S. M. Furnell, "Authentication of users on mobile telephones–a survey of attitudes and practices," *Computers & Security*, vol. 24, no. 7, pp. 519–527, 2005.

[21] R. Amin, T. Gaber, G. ElTaweel, and A. E. Hassanien, "Biometric and traditional mobile authentication techniques: Overviews and open issues," in *Bio-inspiring cyber security and cloud services: trends and innovations*. Springer, 2014, pp. 423–446.

[22] H. Crawford and K. Renaud, "Understanding user perceptions of transparent authentication on a mobile device," *Journal of Trust Management*, vol. 1, no. 1, pp. 1–28, 2014.

[23] T. J. Neal and D. L. Woodard, "Surveying biometric authentication for mobile device security," *Journal of Pattern Recognition Research*, vol. 1, no. 74-110, p. 4, 2016.

[24] H. Khan and U. Hengartner, "Towards application-centric implicit authentication on smartphones," in *Proceedings of the 15th Workshop on Mobile Computing Systems and Applications*, 2014, pp. 1–6.

[25] C. A. Miles and J. P. Cohn, "Tracking prisoners in jail with biometrics: An experiment in a navy brig," *National Institute of Justice Journal*, vol. 253, 2006.

[26] L. Jiang, Z. Zhou, T. Leung, L.-J. Li, and L. Fei-Fei, "Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels," in *International Conference on Machine Learning*. PMLR, 2018, pp. 2304–2313.

[27] Y. Li, J. Yang, Y. Song, L. Cao, J. Luo, and L.-J. Li, "Learning from noisy labels with distillation," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 1910–1918.

[28] J. Xu, Y. Li, and R. H. Deng, "Differential training: A generic framework to reduce label noises for android malware detection," in *Proceedings 2021 Network and Distributed System Security Symposium*. Internet Society, 2021, pp. 1–14.

[29] J. E. Trost, "Statistically nonrepresentative stratified sampling: A sampling technique for qualitative studies," *Qualitative sociology*, vol. 9, no. 1, pp. 54–57, 1986.

[30] G. Chevalier, "Lstms for human activity recognition," 2016, https://github.com/guillaume-chevalier/LSTM-Human-Activity-Recognition.

[31] "TensorFlow," https://www.tensorflow.org/api_docs/python/tf/keras/callbacks/EarlyStopping.

[32] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[33] J. Goldberger and E. Ben-Reuven, "Training deep neural-networks using a noise adaptation layer," 2016.

[34] "PyOD," https://pyod.readthedocs.io/en/latest/.

[35] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah, "Signature verification using a" siamese" time delay neural network," *Advances in neural information processing systems*, vol. 6, pp. 737–744, 1993.

[36] L. Fei-Fei, R. Fergus, and P. Perona, "One-shot learning of object categories. pattern analysis and machine intelligence," *IEEE Transactions on*, vol. 28, no. 4, pp. 594–611, 2006.

[37] S. Chopra, R. Hadsell, and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1. IEEE, 2005, pp. 539–546.

[38] G. Koch, R. Zemel, and R. Salakhutdinov, "Siamese neural networks for one-shot image recognition," in *ICML deep learning workshop*, vol. 2. Lille, 2015.

[39] B. Lake, R. Salakhutdinov, J. Gross, and J. Tenenbaum, "One shot learning of simple visual concepts," in *Proceedings of the annual meeting of the cognitive science society*, vol. 33, no. 33, 2011.

[40] J. Bonneau, C. Herley, P. C. Van Oorschot, and F. Stajano, "The quest to replace passwords: A framework for comparative evaluation of web authentication schemes," in *2012 IEEE symposium on security and privacy*. IEEE, 2012, pp. 553–567.

[41] D. Wang, Q. Gu, H. Cheng, and P. Wang, "The request for better measurement: A comparative evaluation of two-factor authentication schemes," in *Proceedings of the 11th ACM on Asia conference on computer and communications security*, 2016, pp. 475–486.

[42] D. Wang and P. Wang, "Two birds with one stone: Two-factor authentication with security beyond conventional bound," *IEEE transactions on dependable and secure computing*, vol. 15, no. 4, pp. 708–722, 2016.

[43] Q. Wang, D. Wang, C. Cheng, and D. He, "Quantum2fa: efficient quantum-resistant two-factor authentication scheme for mobile devices," *IEEE Transactions on Dependable and Secure Computing*, 2021.

[44] P. Laka and W. Mazurczyk, "User perspective and security of a new mobile authentication method," *Telecommunication Systems*, vol. 69, no. 3, pp. 365–379, 2018.

[45] A. Kruzikova, L. Knapova, D. Smahel, L. Dedkova, and V. Matyas, "Usable and secure? user perception of four authentication methods for mobile banking," *Computers & Security*, vol. 115, p. 102603, 2022.

[46] "System usability scale," https://www.usability.gov/how-to-and-tools/methods/system-usability-scale.html.

[47] S. Trewin, C. Swart, L. Koved, J. Martino, K. Singh, and S. Ben-David, "Biometric authentication on a mobile device: a study of user effort, error and task disruption," in *Proceedings of the 28th Annual Computer Security Applications Conference*, 2012, pp. 159–168.

[48] I. Stylios, S. Kokolakis, O. Thanou, and S. Chatzis, "Behavioral biometrics & continuous user authentication on mobile devices: A survey," *Information Fusion*, vol. 66, pp. 76–99, 2021.

[49] C. E. Shannon, "A mathematical theory of communication," *The Bell system technical journal*, vol. 27, no. 3, pp. 379–423, 1948.

[50] L. F. Kozachenko and N. N. Leonenko, "Sample estimate of the entropy of a random vector," *Problemy Peredachi Informatsii*, vol. 23, no. 2, pp. 9–16, 1987.

[51] D. Wang, Q. Gu, X. Huang, and P. Wang, "Understanding human-chosen pins: characteristics, distribution and security," in *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, 2017, pp. 372–385.

[52] D. Wang, H. Cheng, P. Wang, X. Huang, and G. Jian, "Zipf's law in passwords," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 11, pp. 2776–2791, 2017.

[53] K. Inthavisas and D. Lopresti, "Secure speech biometric templates for user authentication," *IET biometrics*, vol. 1, no. 1, pp. 46–54, 2012.

[54] Y. C. Feng and P. C. Yuen, "Binary discriminant analysis for generating binary face template," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 2, pp.

613–624, 2011.

[55] K. Sadeghi, A. Banerjee, J. Sohankar, and S. K. Gupta, "Geometrical analysis of machine learning security in biometric authentication systems," in *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*. IEEE, 2017, pp. 309–314.

[56] M. Ehatisham-ul Haq, M. A. Azam, U. Naeem, Y. Amin, and J. Loo, "Continuous authentication of smartphone users based on activity pattern recognition using passive mobile sensing," *Journal of Network and Computer Applications*, vol. 109, pp. 24–35, 2018.

[57] Y. Zhang, W. Hu, W. Xu, C. T. Chou, and J. Hu, "Continuous authentication using eye movement response of implicit visual stimuli," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 1, no. 4, pp. 1–22, 2018.

[58] M. Abuhamad, T. Abuhmed, D. Mohaisen, and D. Nyang, "Autosen: Deep-learning-based implicit continuous authentication using smartphone sensors," *IEEE Internet of Things Journal*, vol. 7, no. 6, pp. 5008–5020, 2020.

[59] W.-H. Lee and R. B. Lee, "Multi-sensor authentication to improve smartphone security," in *2015 International conference on information systems security and privacy (ICISSP)*. IEEE, 2015, pp. 1–11.

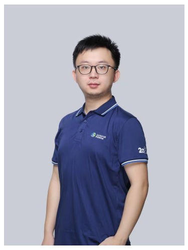
[60] Y. Li, H. Hu, and G. Zhou, "Using data augmentation in continuous authentication on smartphones," *IEEE Internet of Things Journal*, vol. 6, no. 1, pp. 628–640, 2018.

[61] C. Song, A. Wang, K. Ren, and W. Xu, "Eyeveri: A secure and usable approach for smartphone user authentication," in *IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications*. IEEE, 2016, pp. 1–9.

Jie Ying received the B.E. degree in digital media from Zhejiang University of Technology, Hangzhou, China, in 2016. He is currently pursuing Ph.D. degree for computer science in Zhejiang University of Technology. His current research interests include mobile security and system security.

Tiantian Zhu received the Ph.D. degree in computer science from Zhejiang University, Hangzhou, China, in 2019. He is currently an associate professor with the college of computer science and technology, Zhejiang University of Technology, China. His research interests include mobile security, system security and artificial intelligence.

Qiang Liu received the Bacheleor of Science degree in electronic information engineering at Beijing Institute of Technology, Beijing, China, in 2018. He is currently a Ph.D. student in College of Computer Science of Zhejiang University, Hangzhou, China. His research interests include system security, software security and firmware analysis.



Chunlin Xiong received the PhD. on Cybersecurity from Zhejiang University, Hangzhou, China, in 2021. He is currently a postdoctoral student jointly trained by Shenzhen Institute of advanced technology, Chinese Academy of Sciences, and Sangfor Inc.. His research interests include system and information security.

Zhengqiu Weng received the the Ph.D. degree in computer science and technology with the Zhejiang University of Technology, and also a Professor with Wenzhou University of Technology, Zhejiang, China. Her current research interests include networks security and mobile security.

Tieming Chen received the Ph.D. degree in computer software and theory from BeiHang University, Beijing, China, in 2011. He is now a professor with the college of computer science and technology, Zhejiang University of Technology, Hangzhou, China. He is also a member of IEEE and ACM. His research interests include cyberspace security and intelligence security.

Lei Fu received the Ph.D. degree in mechanical engineering from Zhejiang University, Hangzhou, China, in 2018. He is currently a Lecturer with the College of Mechanical Engineering, Zhejiang University of Technology, Hangzhou. His research interests include fault diagnosis, signal processing, and artificial intelligence.

Mingqi Lv received the Ph.D. degree in computer science from Zhejiang University, Hangzhou, China, in 2012. He is currently an associated professor with the college of computer science and technology, Zhejiang University of Technology, China. His research interests include spatiotemporal data mining and mobile security.

Han Wu received the B.E. degree in network engineering from Anhui Normal University, WuHu, China, in 2022. He is currently pursuing M.S. degree for computer science in Zhejiang University of Technology. His current research interests include cyberspace security.

Ting Wang received the Ph.D. degree in computer science from Zhejiang University, Hangzhou, China, in 2015. She is currently an associate professor with the college of computer science and technology, Zhejiang University of Technology, China. Her research interests include system security, formal methods and artificial intelligence.

Yan Chen received the Ph.D. degree in computer science from the University of California, Berkeley, CA, USA, in 2003. He is a Professor with the Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL, USA. Based on Google Scholar, his papers have been cited over 7000 times and his h-index is 34. His research interests include network security, measurement, and diagnosis for large-scale networks and distributed systems. Prof. Chen won the Department of Energy (DoE) Early CAREER Award in 2005, the Department of Defense (DoD) Young Investigator Award in 2007, and the Best Paper nomination in ACM SIGCOMM 2010.