Continuous Authentication of Smartphones Based on Screen-Touch Trajectories

Guozhu Zhao[1], Yuan Jiao[1], Jianyu Zhuo[1], Yu Chen[1], Chuanzhen Ju[1], and Yaxin Wang[1]
[1]The School of Computer and Information Engineering, Chuzhou University, Chuzhou, Anhui 239000, China

As smartphones become essential tools for daily activities, security concerns, particularly in mobile payments and personal privacy, are becoming increasingly critical. Traditional lock screens and password-based authentication systems are vulnerable to various threats, such as unauthorized access and identity theft. To address these issues, this paper proposes an advanced continuous authentication method that verifies user identity based on unique screen-touch trajectories. Using machine learning algorithms, the authentication system analyzes real-time screen-touch trajectories from user interactions with the touchscreen, capturing behavioral patterns that are highly specific to individual users. This proposed continuous authentication framework enables the system to authenticate users seamlessly and continuously throughout device usage, ensuring ongoing protection against unauthorized access, financial fraud, and privacy breaches. Experimental results demonstrate that the Random Forest algorithm outperforms other methods in terms of recognition accuracy and response efficiency, offering a reliable and scalable solution. The paper also explores the integration of this authentication approach into existing smartphone security frameworks, showcasing its potential to significantly enhance network security, particularly in high-risk applications such as mobile payments and personal data access.

***Index Terms*—Continuous authentication, mobile payments and personal privacy, screen-touch trajectories, smartphone security frameworks, payment security.**

I. Introduction

With the widespread adoption of smartphones, mobile payments and personal data security have become critical concerns in modern society. Traditional password-based protection methods are increasingly inadequate in the face of sophisticated, premeditated attacks, particularly in cases where unauthorized access to personal devices occurs. To address these challenges, this paper proposes a continuous authentication system based on screen-touch trajectories. This system aims to mitigate risks associated with financial fraud and breaches of personal privacy, especially when lock screen passwords or payment credentials are compromised [1], [2]. Leveraging machine learning, our system continuously monitors and analyzes users' touch behavior in real time to verify the legitimate user, thereby preventing unauthorized access and financial fraud.

In recent research, user authentication using behavioral biometrics has gained significant attention. Some studies utilize smartphone sensor data for authentication, while others explore the use of acoustic fingerprints as a second factor in multi-factor authentication systems [3]. However, these approaches often depend on specific hardware configurations or are vulnerable to attacks. Moreover, many fail to provide continuous, dynamic identity verification in real-world environments [4].

This paper introduces an innovative continuous authentication method that uses users' screen-touch trajectories as a dynamic, context-aware authentication factor [5]. Unlike traditional methods that rely on static factors such as passwords or biometrics, our approach capitalizes on the inherent variability and unpredictability of user behavior. By analyzing touch trajectories, we generate a constantly evolving authentication context, which poses a significant challenge for attackers attempting to replicate or predict the user's behavior [6]. This dynamic approach adds a robust layer of security while aligning with user-friendly practices, avoiding the disruptions often associated with conventional user-centric authentication mechanisms. Additionally, it minimizes the need for extra hardware investments by utilizing existing smartphone sensors and data analytics capabilities [7].

The main contributions of this paper are as follows:

- We propose a novel continuous authentication method that combines screen-touch trajectory features to form a unique feature space, using user behavior patterns as an authentication factor. This approach provides continuous, context-aware authentication, enabling the detection of unauthorized access during user interactions.
- We developed and evaluated several ensemble classifiers, including K-Nearest Neighbors (KNN) [8], Support Vector Machines (SVM) [9], eXtreme Gradient Boosting (XGBoost) [10], and Random Forest [11] to assess the legitimacy of user interactions with mobile devices. Through extensive experimentation, we demonstrate the effectiveness of our proposed framework in verifying user authenticity.

II. PROBLEM FORMULATION

A. System Model

In the field of information security, particularly with regard to personal privacy and asset protection, continuous authentication plays a critical role in environments that require high levels of privacy. In our system model, we propose a continuous authentication method based on users' screen-touch trajectories to bolster the security of personal devices against unauthorized access. We hypothesize that users' touch trajectories are unique to individuals, similar to signatures or

Manuscript received November 14, 2024; revised December 23, 2024.
Corresponding author: Yuan Jiao (email: 1469675134@qq.com).

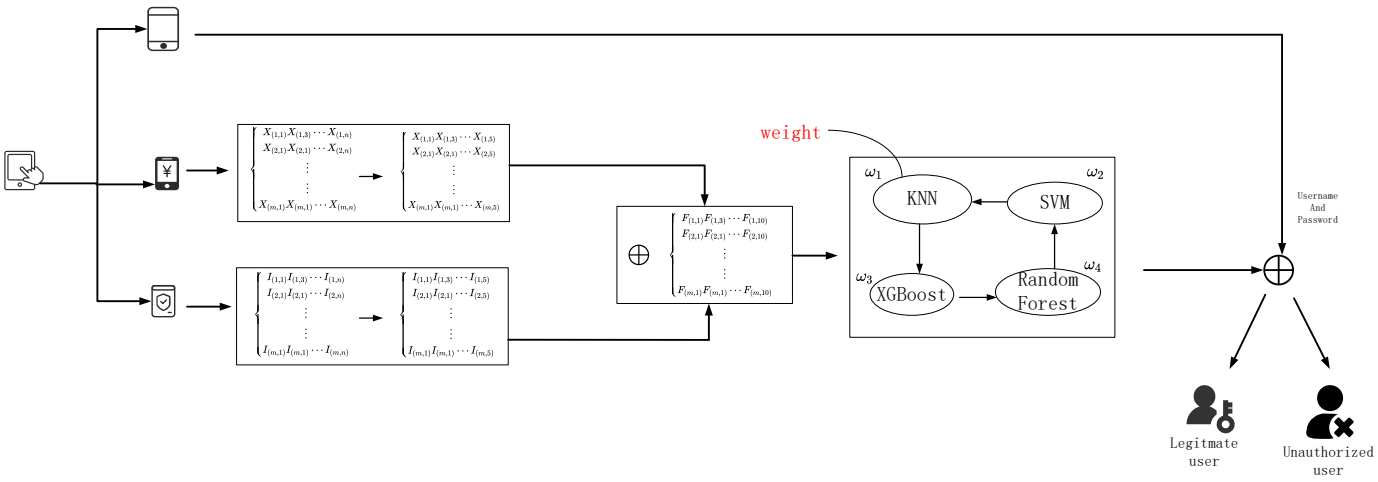

Fig. 1. Performance of ensemble classifiers (the weights of the two classifiers are 0.5)

fingerprints. As such, when a user attempts to unlock their device or perform payment transactions, the system captures their screen-touch trajectories and compares them to pre-stored reference data. If significant deviations are detected, the system will flag the interaction as potentially unauthorized, triggering security measures such as locking the device or issuing an alert.

By integrating behavioral characteristics into the authentication process, our approach aims to offer a more secure and reliable protection mechanism for personal devices, effectively countering increasingly sophisticated security threats. This method not only strengthens device security but also provides a solution that does not require additional hardware, aligning with the practical needs of modern mobile device usage.

B. Threat Model

In this paper, we examine attack scenarios where adversaries attempt to authenticate by mimicking legitimate user behavior. For instance, an attacker may try to replicate a user's screen-touch trajectories or reduce the number of swipes in an attempt to evade detection. Such actions could potentially expose sensitive information, including financial data and personal details, to the attacker. Our objective is to design an authentication mechanism with high recognition capabilities and sensitivity, capable of detecting attackers based on discrepancies in their touch trajectories.

In this model, attackers unknowingly undergo implicit identity verification during their interactions with the device. The results of this verification are then reported to the operating system, which executes the appropriate security response. We specifically focus on authenticating the legitimate owner of a smartphone in personal device scenarios, where the device is typically used exclusively by its owner and not shared.

The following attack scenarios are considered in our threat model:

- Imitation of Screen-touch Trajectories: An attacker may observe or record a user's typical screen interactions and attempt to replicate their swiping patterns in order to deceive the authentication system.
- Reduction of Swipe Counts: An attacker may deliberately minimize the number of swipes to decrease the likelihood of triggering the continuous authentication system, thereby avoiding detection.

III. CONTINUOUS AUTHENTICATION FRAMEWORK

The proposed continuous authentication framework, based on screen-touch trajectories, is designed to enhance the security of personal mobile devices. By analyzing users' behavioral characteristics, this framework can accurately identify the legitimate device owner, effectively preventing unauthorized access and financial fraud [12]. The framework consists of three primary stages: 1) Feature collection and extraction. 2) Feature processing and integration. 3) Continuous authentication. A detailed overview of the framework is presented in Figure 1.

A. Feature Collection and Extraction

The uniqueness of individual screen-touch behaviors provides a novel approach to identity authentication. To leverage this, we employ Android Debug Bridge (ADB) wireless technology, in combination with the built-in sensors of smartphones, to accurately capture users' screen-touch trajectories. When users unlock their phones or perform payment transactions, we collect these trajectory data in real-time through a computer console, using them as key features for identity verification.

The data collection process proceeds as follows. First, ensure that both the smartphone and computer are connected to the same local area network. Next, on the computer console, execute the ADB pairing command, input the smartphone's IP address, port number, and a preset pairing code to establish a stable wireless connection. Once the connection is established, use the ADB shell command to access the phone's terminal and execute the getevent -ltr command to capture real-time event data from the phone, including screen touches, touch

pressure, and timestamps. Finally, the collected data is organized and saved into a text file on the computer, which forms the basis for subsequent feature extraction.

In the feature extraction phase, we focus on the spatiotemporal characteristics of the screen-touch trajectories, including touch coordinates, pressure levels, and timestamps—key elements that accurately capture the uniqueness of user behavior. These features are then represented mathematically within a feature matrix, forming a comprehensive feature set that encapsulates rich behavioral context. This feature set provides robust data support for the next steps in the identity authentication process.

B. Feature Processing and Combination

After preprocessing the payment and privacy experiment data, we extracted a five-dimensional feature matrix from the raw data. Specifically, four key features were extracted from the experimental data, each containing M data points. These features are:

- The set of X-axis coordinates of touch points $\{X_1X_2\cdots X_M\}$
- The set of Y-axis coordinates of touch points $\{Y_1Y_2\cdots Y_M\}$
- The set of touch timestamps $\{T_1T_2\cdots T_M\}$
- The set of touch pressure values $\{Pr_1Pr_2\cdots Pr_M\}$

Using the X and Y coordinates of the M touch points, we calculated the slide trajectory lengths between adjacent points using the Euclidean distance formula. Specifically, for each pair of consecutive points (X_n, Y_n) and (X_{n+1}, Y_{n+1}), the length of screen-touch trajectory D_n is given by

$$D_n = \sqrt{(X_n - X_{n+1})^2 + (Y_n - Y_{n+1})^2}. \tag{1}$$

This process generates the set of slide trajectory lengths $\{D_1D_2\cdots D_{M-1}\}$. From this, we can compute the average length of screen-touch trajectory $\bar{D}$ as

$$\bar{D} = \frac{\sum_{n=1}^{M-1} D_n}{M-1}. \tag{2}$$

Additionally, the maximum and minimum lengths of screen-touch trajectories in this set are denoted as max(D) and min(D) , respectively.

Next, using the timestamps of adjacent touch points and their corresponding lengths of screen-touch trajectories, we calculate the speed of each length of screen-touch trajectory. Assuming the touch time of the previous point is T_n, the touch time of the current point is T_{n+1}, and the length of screen-touch trajectory is D_n, the speed Sp_n is given by

$$Sp_n = \frac{D_n}{(T_{n+1} - T_n)}. \tag{3}$$

This produces the set of length of screen-touch trajectory speeds $\{Sp_1Sp_2\cdots Sp_{M-1}\}$, and the average speed $\bar{Sp}$ is calculated as

$$\bar{Sp} = \frac{\sum_{n=1}^{M-1} Sp_n}{M-1}. \tag{4}$$

Finally, the average touch pressure $\bar{Pr}$ is calculated by averaging the set of touch pressure values

$$\bar{Pr} = \frac{\sum_{n=1}^{M} Pr_n}{M}. \tag{5}$$

This results in a five-dimensional feature matrix containing the following features: $\bar{D}$, min(D), max(D), $\bar{Sp}$, and $\bar{Pr}$.

To construct a combined feature space for training, we introduce S_n, a feature space containing N samples. Here, F_a represents the features extracted from the payment experiment, and F_b represents the features extracted from the privacy experiment. The combined feature space is represented as

$$S_N = \{[F_{a_1}, F_{b_1}], [F_{a_2}, F_{b_2}], \cdots, [F_{a_N}, F_{b_N}]\}. \tag{6}$$

As shown in Equation (6), the two domain feature sets of size (5,N) are merged to form a feature matrix of size (10,N). This process preserves a relatively small feature dimension while maintaining the integrity of the environmental features, providing effective input for subsequent model training.

C. Classifier Training

In smartphone systems, user authentication mechanisms can be viewed as classification tasks. To accurately distinguish legitimate user behavior or perform other classification tasks, we employ four widely used training methods: KNN, SVM, XGBoost, and Random Forest. Each of these methods is used to construct a classification model based on screen-touch trajectories.

For the feature matrix described above, we define S_N as the feature space containing N samples. Each sample consists of features from both the payment experiment F_a and the privacy experiment F_b , represented as $S_N = \{[F_{a_1}, F_{b_1}], [F_{a_2}, F_{b_2}], \cdots, [F_{a_N}, F_{b_N}]\}$.

To train the classification models, we extract a subset S_m containing m samples from S_N, where m is an integer between 1 and N, represented as

$$S_N = \{[F_{a_1}, F_{b_1}], [F_{a_2}, F_{b_2}], \cdots, [F_{a_m}, F_{b_m}]\}. \tag{7}$$

1) K-Nearest Neighbors: KNN is an instance-based learning algorithm that classifies a sample based on the majority vote of its k closest neighbors in the feature space. For a new sample q, its feature vector is a 10-dimensional vector, matching the dimensionality of the feature matrix rows. The Euclidean distance $E(p, q)$ between q and a sample p in S_m is calculated using

$$E(p, q) = \sqrt{\sum_{i=1}^{10} (p_i - q_i)^2} \tag{8}$$

where p_i and q_i are the feature values of samples p and q respectively. After computing the distances, the k closest samples are selected, and the most frequent label among them is assigned to q. If the label for legitimate users is most frequent, q is classified as a legitimate user; otherwise, it is assigned to another category.

2) Support Vector Machine: SVM classifies by finding the optimal separating hyperplane with the largest margin

in the feature space S_m. The separating hyperplane is defined by the linear equation

$$\xi^T x + s = 0 \tag{9}$$

where ξ is the normal vector determining the hyperplane's direction, and s is the displacement term that determines the hyperplane's distance from the origin. The objective of SVM is to maximize the margin γ between different classes, defined by

$$\gamma = \frac{2}{||\xi||}. \tag{10}$$

To achieve this, we minimize the squared norm of the normal vector

$$-\frac{||\xi||^2}{2}. \tag{11}$$

While ensuring the constraints

$$y_i(\xi^T x + s) \geq 1, for\ i = 1, 2, \ldots, m. \tag{12}$$

To ensure all samples are correctly classified.
Where y_i is the label of sample i, and x is a sample in S_m. The support vectors are those closest to the hyperplane and play a key role in determining its position. Through optimization techniques such as the Lagrange multiplier method or Sequential Minimal Optimization (SMO), we can find the optimal hyperplane (ξ^*, s^*).
Once trained, SVM classifies new samples by evaluating

$$\xi^{*T} x + s^* = 0. \tag{13}$$

A positive result indicates class +1, while a negative result indicates class 1.

3) eXtreme Gradient Boosting: XGBoost is a gradient boosting framework that builds an ensemble of decision trees (CART trees) in an iterative manner. The objective function for XGBoost is defined as

$$L(\varphi) = \sum_{i=1}^{n} l(y_i, \hat{y_i}) + (f_i) \tag{14}$$

where $l(y_i, \hat{y_i})$ is the loss function, and (f_i) is a regularization term that prevents overfitting. In each iteration, XGBoost calculates the prediction residuals and builds a new tree to correct these residuals, refining the model's prediction. The prediction for a new sample is the sum of the outputs of all decision trees in the ensemble.
XGBoost optimizes the objective function using second-order Taylor expansion and differentiation to find the optimal decision tree structure and leaf predictions, minimizing the loss function until a stopping criterion is met, such as reaching the maximum number of iterations or achieving no improvement.

4) Random Forest: Random Forest is an ensemble learning method that constructs multiple decision trees to improve classification accuracy and robustness. It generates multiple bootstrap samples from the original dataset S_N, with each sample having the same number of instances as the original dataset. These subsets, referred to as "out-of-bag" samples, are used to train individual decision trees.
For each tree, Random Forest randomly selects a subset of features at each node and identifies the best split. The tree is grown until a stopping criterion, such as the maximum depth or the minimum number of samples, is met. After all trees are trained, predictions for new samples are made through a majority voting mechanism for classification tasks, or by averaging the outputs of all trees for regression tasks.
Random Forest also uses out-of-bag samples for performance evaluation, eliminating the need for separate cross-validation and thus enhancing efficiency.

D. Sensitivity of Classifier Weights

The decision-making process for user authentication can be expressed by Equation (15). Given a user with a claimed identity I and their associated feature set $[F_a, F_b]$, our authentication system utilizes an ensemble classifier H [13]. The decision rule is defined as follows

$$H(I, [f_{an}, f_{fi}]) \in \begin{cases} \varpi_1, if\ \omega_1 P_1(C_1([f_a, f_b])) + \\ \omega_2 P_2(C_2([f_a, f_b])) \geq \theta \\ \varpi_2, otherwise \end{cases} \tag{15}$$

where θ is a predefined threshold. $P_1()$ and $P_2()$ are the outputs (probabilities) of classifiers C_1 and C_2 , respectively. Here, ω_1 corresponds to the scenario where the claimed identity is valid (legitimate user), and ω_2 corresponds to the scenario where the identity is invalid (attacker). Additionally, ω_1 and ω_2 are weights that satisfy the constraint $\omega_1 + \omega_2 = 1$.

IV. EXPERIMENT AND ANALYSIS

A. Data Acquisition and Performance Metrics

To evaluate the performance of our proposed authentication mechanism under real-world smartphone usage conditions, we employed a data collection strategy that incorporates both positive and negative samples. Positive samples refer to data generated by the legitimate user during authorized operations, while negative samples are generated by unauthorized users attempting to perform similar tasks.As presented in Table I.

Experiment 1: Payment Operations.

In the first experiment, we focused on simulating payment operations. We invited 10 different individuals to attempt unauthorized access on the same mobile phone. These individuals were asked to perform 20 sets of payment-related tasks, such as attempting fraudulent transactions. The data collected from these attempts formed Subset 1. These samples represent negative data generated by unauthorized users.Meanwhile, one target user (the legitimate user) conducted 200 sets of legitimate payment operations, interacting with the payment application on the phone. These interactions involved performing real, authorized payment transactions. The data from these operations was collected as Subset 2, representing positive data for the legitimate user.

Experiment 2: Privacy-Related Operations.

In the second experiment, the same 10 individuals were tasked with performing privacy-related operations on the mobile phone, such as attempting to access or modify personal information that did not belong to them. Each of the 10

TABLE I: MAIN EXPERIMENT DATASETS

Dataset Name	Feature Domain	Subjects	Data Acquisition
Dataset 1	Payment	400 data files (200 from legal users; 200 from illegal users; Both performing the same tasks)	Subset 1: 200 sets of data collected from legal users via smartphone. Subset 2: 200 sets of data collected from illegal users via smartphone.
Dataset 2	Privacy	400 data files (200 from legal users; 200 from illegal users; Both performing the same tasks)	Subset 1: 200 sets of data collected from legal users via smartphone. Subset 2: 200 sets of data collected from illegal users via smartphone.

unauthorized users again performed 20 sets of such tasks, resulting in Subset 3, representing negative data generated by unauthorized users in the context of privacy violations. The target user then conducted 200 sets of privacy-related tasks on the phone, including accessing their personal data, modifying settings, and interacting with private information on the device. This data formed Subset 4, representing positive data from the legitimate user in the privacy domain.

To ensure the validity of the results, we conducted thorough statistical analysis on the collected data. The analysis included the following steps:

1) Data Preprocessing: The raw touch trajectory data were cleaned to remove inconsistencies or noise. This involved normalizing touch coordinates, pressure values, and timestamps across all datasets to ensure uniformity.
2) Feature Extraction: Features such as touch coordinates, pressure levels, trajectory length, and timestamps were extracted for each sample. These features were used to construct the feature matrix for each dataset, representing user behavior in both the payment and privacy contexts.
3) Cross-Validation: To evaluate the generalization performance of the authentication mechanism, we performed k-fold cross-validation, where the dataset was split into k subsets. Each subset was used for testing, while the remaining subsets were used for training. This procedure was repeated for each fold to minimize overfitting.
4) Performance Metrics: The following performance metrics were used to assess the effectiveness of the authentication mechanism:
   - Accuracy: The proportion of correctly classified samples (both positive and negative) out of the total samples.
   - Precision: The proportion of true positive samples among all samples classified as positive.
   - Recall: The proportion of true positive samples among all actual positive samples.
5) ROC Curve and AUC: The Receiver Operating Characteristic (ROC) curve was used to evaluate the performance of our authentication mechanism. It plots the trade-off between the True Positive Rate (TPR), also known as sensitivity, and the False Positive Rate (FPR), also known as 1-specificity, at various classification thresholds. The TPR is calculated as

$$TPR = \frac{TP}{(TP + FN)} \tag{16}$$

where TP is the number of true positives, and FN is the number of false negatives. The FPR is calculated as

$$FPR = \frac{FP}{(FP + FN)} \tag{17}$$

where FP is the number of false positives, and TN is the number of true negatives. The ROC curve visualizes the performance of the classifier across different thresholds, showing how well it distinguishes between legitimate and unauthorized users.
The Area Under the Curve (AUC) is a single scalar value that quantifies the overall performance of the classifier. It is the area under the ROC curve, which ranges from 0 to 1. A value of 1 indicates perfect classification, while a value of 0.5 indicates random guessing. The higher the AUC, the better the classifier is at distinguishing between legitimate users and attackers.

In total, 800 data samples were collected across both experiments, with 400 positive samples (from the legitimate user) and 400 negative samples (from unauthorized users). These samples were used to evaluate the authentication mechanism's performance using various classification models. The results were analyzed using standard metrics such as accuracy, precision, recall, and AUC to determine the effectiveness of the proposed authentication approach.

B. Performance Analysis of Individual Classifiers

We combine data from both the payment and privacy experiments, processing and merging the datasets to create a feature set of 400 attributes, comprising 200 positive and 200 negative samples. Seventy percent of the samples are allocated to the training set, with the remaining thirty percent used for testing. Four classifiers—KNN, SVM, XGBoost, and Random Forest, are applied to evaluate performance on the test set. The corresponding ROC curves, which clearly depict the TPR on the y-axis and the FPR on the x-axis, are shown in Figure 2.

As illustrated, SVM and KNN show comparatively lower classification performance, each achieving an AUC of 0.849.

In contrast, Random Forest outperforms XGBoost, with respective AUC values of 0.974 and 0.944, establishing Random Forest as the most effective classifier in this study.

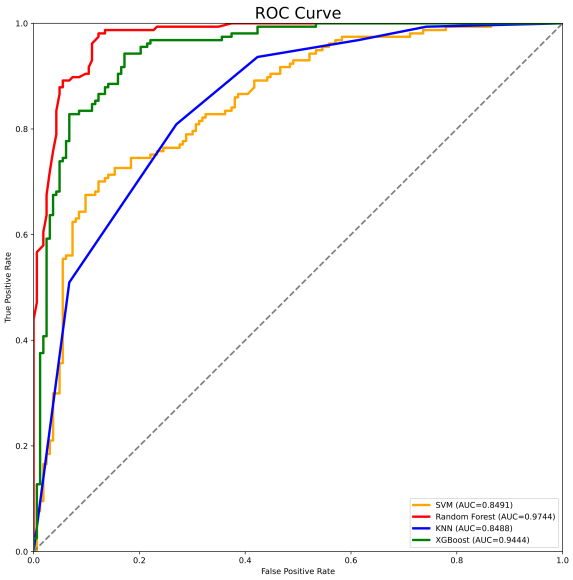

Fig. 2. Performance of classifiers SVM, Random Forest, KNN, XGBoost

C. Performance Analysis of Ensemble Classifiers

We combined data from both the payment and privacy experiments, processing and merging the datasets to create a comprehensive feature set. Utilizing these data, we evaluated the performance of various machine learning algorithms, including SVM, Random Forest, KNN, and XGBoost, as well as their combinations. The ROC curves generated from these evaluations, which clearly depict the TPR against the FPR, are presented in the accompanying Figure 3. Each point on the ROC curves represents a specific model, with the AUC serving as a metric of classification performance.

In the Figure 3, the x-axis represents the FPR and the y-axis represents the TPR. As illustrated, the ensemble classifier combining Random Forest and KNN achieved the best classification performance, with an AUC of 0.9660. This aligns with the individual classifier results, where Random Forest and KNN, when used in combination, outperformed the other classifiers.

D. Sensitivity to Classifier Weights

The experiments conducted in Sections IV-B and IV-C demonstrate that the best classification performance is achieved through the weighted combination of Random Forest and KNN. To further investigate the optimal results, we compared classifier performance across different values of ω_4 within the range [0,1]. To minimize result fluctuations for specific weights, we conducted 10 trials for each ω_4 value, averaged the outcomes, and present the results in Figure 4.

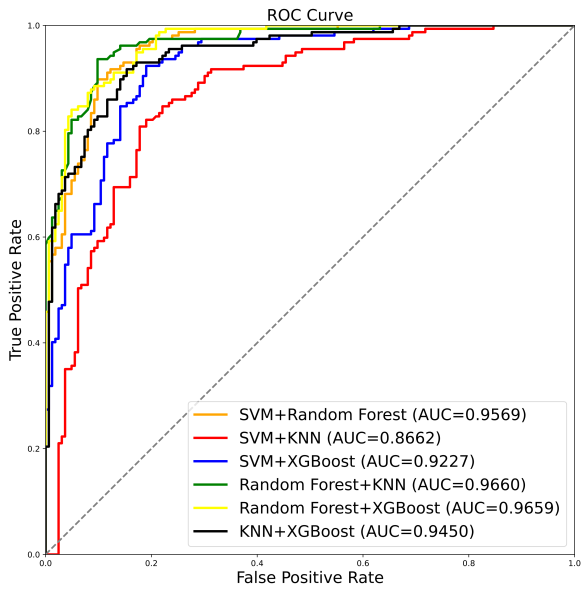

Fig. 3. Performance of ensemble classifiers (the weights of the two classifiers are 0.5)

As shown in Figure 4, the x-axis represents the weight assigned to the Random Forest classifier (ω_1), ranging from 0 to 1. Consequently, the weight assigned to the KNN classifier (ω_4) decreases from 1 to 0 as ω_1 increases. The y-axis indicates the AUC values, ranging from approximately 0.86 to 0.98. When ω_4 = 0.2 (i.e., ω_1 = 0.8), the decision results exhibit significant fluctuations. However, when ω_4 = 0.8 (i.e., ω_1 = 0.2), the ensemble classifier combining Random Forest and KNN performs the best, with a stable AUC of 0.9776. This performance is superior to that of either Random Forest or KNN as standalone classifiers.

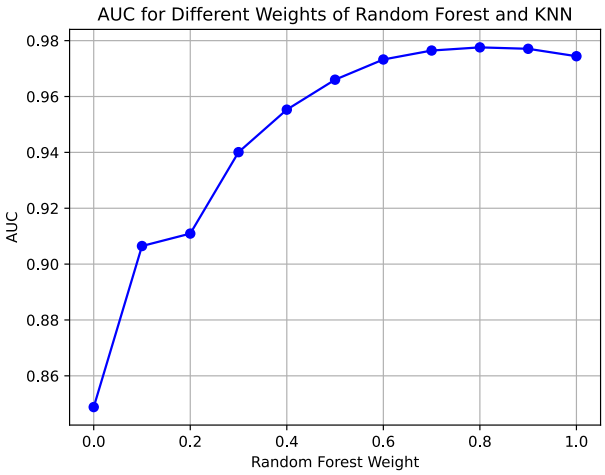

Fig. 4. The weight allocation of classifiers impacts the performance of the ensemble classifier Random Forest +KNN

V. Conclusion

This paper presents a continuous authentication system based on screen touch trajectories to strengthen smartphone security, with a particular focus on safeguarding mobile payments and personal privacy. By addressing the limitations of traditional lock screens and password-based authentication, our approach analyzes user behavior in real time to verify device ownership, effectively preventing unauthorized access and financial fraud. Using machine learning algorithms, the system captures unique swipe patterns, triggering security measures when detecting suspicious behavior. Experimental results show that the Random Forest classifier outperforms other models in both accuracy and response time.

Additionally, we explore integrating this continuous authentication system within smartphone security frameworks to enhance broader network protection. A key innovation of this study lies in applying biometric identification techniques to the analysis of the trajectory of the touchscreen, providing an added layer of security for mobile payments. By continuously monitoring screen-touch behavior, our system ensures secure access without compromising user experience.

Acknowledgment

This work was supported in part by the Anhui Province Outstanding Young Teacher Training Project (YQYB2023048), in part by the New Generation Information Technology Innovation Project of the 2023 China University Industry Research Innovation Fund (2023IT010), and in part by the Chuzhou Science and Technology Plan Project (2023ZD028, 2023ZD029).

References

[1] Saeed Samet, Mohd Tazim Ishraque, Mehdi Ghadamyari, Krishna Kakadiya, Yash Mistry, and Youssef Nakkabi. Touchmetric: a machine learning based continuous authentication feature testing mobile application. *International Journal of Information Technology*, 11(4):625–631, 2019.

[2] Ingo Deutschmann, Peder Nordström, and Linus Nilsson. Continuous authentication using behavioral biometrics. *IT professional*, 15(4):12–15, 2013.

[3] Lei Yang, Yi Guo, Xuan Ding, Jinsong Han, Yunhao Liu, Cheng Wang, and Changwei Hu. Unlocking smart phone through handwaving biometrics. *IEEE Transactions on Mobile Computing*, 14(5):1044–1055, 2014.

[4] Vishal M Patel, Rama Chellappa, Deepak Chandra, and Brandon Barbello. Continuous user authentication on mobile devices: Recent progress and remaining challenges. *IEEE Signal Processing Magazine*, 33(4):49–61, 2016.

[5] Elaine Shi, Yuan Niu, Markus Jakobsson, and Richard Chow. Implicit authentication through learning user behavior. In *Information Security: 13th International Conference, ISC 2010, Boca Raton, FL, USA, October 25-28, 2010, Revised Selected Papers 13*, pages 99–113. Springer, 2011.

[6] Elakkiya Ellavarason, Richard Guest, Farzin Deravi, Raul Sanchez-Riello, and Barbara Corsetti. Touch-dynamics based behavioural biometrics on mobile devices–a review from a usability and performance perspective. *ACM Computing Surveys (CSUR)*, 53(6):1–36, 2020.

[7] Vishal M Patel, Rama Chellappa, Deepak Chandra, and Brandon Barbello. Continuous user authentication on mobile devices: Recent progress and remaining challenges. *IEEE Signal Processing Magazine*, 33(4):49–61, 2016.

[8] Diego Fernández, Vreixo Formoso, Fidel Cacheda, and Victor Carneiro. High order profile expansion to tackle the new user problem on recommender systems. *PloS one*, 14(11):e0224555, 2019.

[9] Bernhard E Boser, Isabelle M Guyon, and Vladimir N Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152, 1992.

[10] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

[11] Rania R Kadhim and Mohammed Y Kamil. Comparison of machine learning models for breast cancer diagnosis. *IAES International Journal of Artificial Intelligence*, 12(1):415, 2023.

[12] Stephen Brewster and Aurora Constantin. Tactile feedback for ambient awareness in mobile interactions. In *Proceedings of HCI 2010*. BCS Learning & Development, 2010.

[13] Giorgio Giacinto, Fabio Roli, and Luca Didaci. Fusion of multiple classifiers for intrusion detection in computer networks. *Pattern recognition letters*, 24(12):1795–1803, 2003.