CAuthNet: A Unified Embeddings for Continuous Authentication in Smartphones

Yogachandran Rahulamathavan, Varuna De Silva, Chaminda Hewage, and Sangarapillai Lambotharan *Senior Member, IEEE*

Abstract—Despite significant recent advances in continuous authentication, implementing it efficiently at scale remains challenging. In this paper, we introduce CAuthNet, a system that directly learns a mapping from smartphone sensor readings related to touch events to a compact Euclidean space, where distances correspond to a measure of touch similarity on the phones. We train one-shot Seismic network architecture tailored for continuous authentication. This approach streamlines continuous authentication by training a single model using training data belonging to a few users. However, the trained model can then be deployed on new users' smartphones without further training, ensuring both privacy and portability. As the smartphone sensors' data is often noisy, we introduced a novel feature extraction technique for each touch events and shown that aggregating few consecutive touches significantly enhances the accuracy of the trained model. The proposed algorithm is validated using the H-MOG dataset, one of the largest publically available smartphone datasets, which contains an accelerometer, gyroscope, magnetometer and touch sensor data collected from 100 users. We used half of the users' data to train a model and then tested the performance of the trained model using the remaining half of the users. The proposed algorithm achieves up to 97% true positive and true negative rates (3% Equal Error Rate) on unseen users. This is the first known portable continuous authentication technique with a simple and scalable architecture, making it suitable for real-world deployments.

Index Terms—Continuous Authentication, One-shot Learning, and Anomaly Detection.

1 Introduction

We now heavily rely on smartphone applications to access a wide variety of services ranging from banking, emails, and social media to healthcare, online shopping, and entertainment. In the early days, these applications were protected by smartphone authentication methods such as passwords and PINs. The passwords and PINs can be easily forgotten or stolen and not very user-friendly. Recently, modern smartphones have adopted biometric methods such as face recognition, fingerprint scanning, and iris detection to authenticate users and provide secure access to the device and sensitive information. Additionally, many sensitive applications now require extra authentication each time users access them, enhancing security measures beyond the initial device unlock. However, these authentication mechanisms typically only verify the user at the point of entry, not continuously during the usage of applications, leaving potential security gaps during active sessions. Moreover, they require users to stop what they are doing and actively enter credentials, disrupting the user experience and potentially reducing the overall convenience and effectiveness of the security measures.

This is where continuous authentication comes in. Continuous authentication passively monitors the users' behavioural biometrics while using their phone, analyzing things like how users type, swipe, and hold the device [1], [2]. These unique behavioral patterns are used to continuously verify the users identity in the background, without interrupting their workflow. Continuous authentication provides an added layer of security beyond the initial login to applications, making it significantly more difficult for someone to hijack an application after the initial authentication due to the challenge of mimicking the user's unique behavior patterns. Additionally, it enhances the user experience by eliminating the need for users to repeatedly enter their credentials, ensuring seamless and secure access throughout the session. Commonly used behavioural patterns in continuous authentication include keystroke dynamics, touch behaviour patterns [6], and motion behaviour patterns [7]. Among these, touch behaviour biometrics have been extensively studied and demonstrate the most promising potential for continuous authentication [16].

Continuous authentication integrates behavioral biometrics with advanced machine learning algorithms to continuously scrutinize user activity for signs of unauthorized access. Behavioral biometrics encompass individualized patterns such as typing cadence, touch patterns, and scrolling behavior. These patterns are modeled using smartphone sensor data from accelerometers, gyroscopes, magnetometers, touch sensors, vibration, and orientation. To build a highly reliable continuous authentication system, a robust machine learning model must continuously monitor and analyze user behavior, comparing it with past interactions. The model starts by capturing the user's baseline behavior during the login phase and the commencement of the session, establishing a reference point for subsequent evaluations.

- *Y. Rahulamathavan, V. Silva, and S. Lambotharan are with the Institute for Digital Technologies, Loughborough University London, London, U.K. (e-mails: {y.rahulamathavan, V.D.De-Silva, s.lambotharan}@lboro.ac.uk).*
- *C. Hewage is with the Cardiff Metropolitan University, U.K. (e-mail: chewage@cardiffmet.ac.uk).*

- *Source code is available at this GitHub repository: [Python] https://github.com/rahulay1/CAuthNet and [Android] https://github.com/rahulay1/CAuthNet_Android/.*

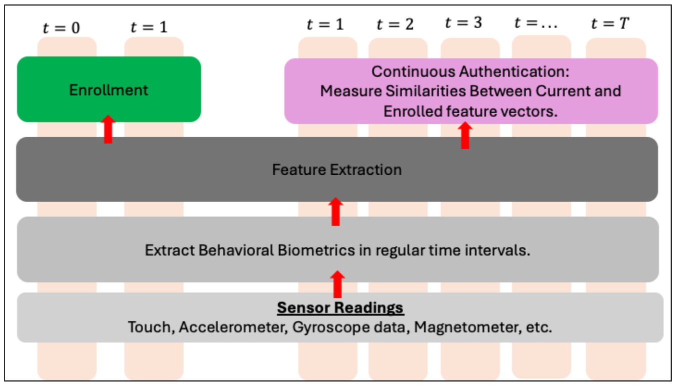

$t = 0$
$t = 1$
$t = 1$
$t = 2$
$t = 3$
$t = \ldots$
$t = T$
Enrollment
Continuous Authentication: Measure Similarities Between Current and Enrolled feature vectors.
Feature Extraction
Extract Behavioral Biometrics in regular time intervals.
Sensor Readings
Touch, Accelerometer, Gyroscope data, Magnetometer, etc.

Figure 1: Key components of any continuous authentication algorithm.

Over time, the machine learning model periodically assesses the user's actions and patterns, comparing them against the established baseline to detect any deviations or anomalies. In this paper, we propose using one-shot learning based on a seismic neural network architecture to develop such a machine learning model [25]. Unlike traditional methods requiring extensive training data, one-shot learning based on a seismic neural network enables models trained on one set of users to be deployed seamlessly to new users without being required to retrain the model.

2 Related Works

2.1 Gait Analysis

At a high level, continuous authentication can be classified into two categories: (1) using gait analysis with wearable sensors to authenticate users, and (2) authenticating users based on their interactions with smartphones to protect applications. Gait analysis involves monitoring and analyzing the unique walking patterns of individuals, with sensors typically placed on various parts of the body, such as the hand, hip, or pockets, to capture these patterns [3]–[5]. While gait-based authentication schemes offer high accuracy due to their ability to capture distinctive body movements [7], the requirement for additional wearable sensors can be inconvenient. Since our work falls into the second category, we review relevant state-of-the-art works for authenticating users based on their interactions with smartphones.

2.2 Smartphone sensor-based solutions

Since Frank et al. [6] first studied in 2012 whether smartphone sensors could be exploited to continuously authenticate users, numerous works have proposed various algorithms that use different combinations of sensors, machine learning algorithms, and feature extraction techniques, as shown in Table 1 [6]–[22]. Table 1 compares the key characteristics of these state-of-the-art works.

The third column in Table 1 denotes the sensors used by the respective works, where To, Ac, Gy, Ma, Or, Loc, App, Vi denote touch, acceleration, gyroscope, magnetometer, orientation, location, app usage, and vibration sensors, respectively. The usage of motion sensors (accelerometer, gyroscope, and magnetometer) is popular among most of the works. However, [6], [15] use only touch sensors, and [20] uses vibration sensors.

The fifth column denotes the machine learning algorithms used by the respective works, where k-NN, SVM, NN, HMM, LSTM, CNN, GAN, TF, GBDT, RNN, PCA, EE, OC, and DT denote k-Nearest Neighbors, Support Vector Machine, Neural Network, Hidden Markov Model, Long Short-Term Memory, Convolutional Neural Network, Generative Adversarial Network, Transformers, Gradient Boosting Decision Tree, Recurrent Neural Network, Principal Component Analysis, Elliptic Envelope, One-Class Classification, and Decision Tree, respectively.

As indicated in the accuracy column in Table 1, all of these works achieve an Equal Error Rate (EER) in the range of 0.1% to 9%, demonstrating their effectiveness in continuous authentication. However, we can split these works into two categories: classification-based methods [6], [9]–[12], [14]–[20], [22], which achieve very low EER, and anomaly detection-based methods [7], [8], [13], [19], [21], which typically have slightly higher EER. Below, we demonstrate why these schemes are not scalable.

2.3 Classification Approches

In this section, we describe the classification-based continuous authentication schemes proposed in the literature, where data from multiple users are required to train a user-specific model. Frank et al. utilized touchscreen input as a behavioural biometric, asking 41 subjects to read documents or view images on smartphones. Their study exploited SVM and kNN classifiers to obtain models with EERs of less than 3% for 11 touch gestures [6]. CAGANet, presented in [11], collects accelerometer, gyroscope, and magnetometer data while a user operates a smartphone. The data is preprocessed and augmented, with CNN extracting features. PCA selects discriminative features, and classifiers such as OC-SVM, LOF, IF, and EE are utilized for authentication to achieve EER of less than 4%.

In [12], Kalman filtering and wavelet techniques were used to minimise the impact of noise from sensors data. Then the work [12] used singular value decomposition for dimensionality reduction for discriminability and stability of intrinsic features from sequential operation actions. The authentication decision process was formulated as a HMM using one-class classification, constructing unique user operation-action characteristics over time and achieved EER of less than 5%.

To address the challenges of insufficient training data and inefficient feature combination, [14] explored a transformer-based GAN to generate additional data for CNN training. An adaptive-weighted concatenation method was designed for CNN-extracted feature fusion. Using an OC-SVM classifier with RBF kernel function, their approach achieved EER of around 0.1% with data from 10 volunteers.

CT-Auth [15] leverages raw capacitive values collected from a smartphone's capacitive touchscreen for authentication based on touch behavior. The framework uses a three-dimensional CNN to capture intra-gesture spatial-temporal features and a structure extraction model for capturing structural information between moving fingertips and the

Table 1: SUMMARY OF THE RELATED WORKS.

Literature	**Year**	**Sensor**	**Biometrics**	**Machine Learning Algo.**	**Dataset**	**# of Users**	**Accuracy**	**Novelty**	**Privacy**	**Portable**
Frank [6]	2012	To	Swipe	k-NN SVM	Public	41	EER <3% EER <3%	Multi-Touch	✗	✗
HMOG [7]	2015	To, Ac, Gy, Ma	Motion	SVM	H-MOG	100	EER <9%	Features Multi-Touch	✓	✗
Shen [8]	2017	To, Ac, Gy, Ma	Touch	SVM, NN HMM	Private H-MOG	102 100	EER <5% EER <6%	Multi Scenario	✗ ✓	✗ ✗
SensorAuth [9]	2018	Ac, Gy	Motion	SVM	H-MOG	100	EER <5%	Data Aug.	✗	✗
AUTOSen [10]	2020	Ac, Gy, Ma	Motion	LSTM	Private	84	FRR <7%	Implicit	✗	✗
CNN-G [11]	2021	Ac, Gy, Ma	Motion	CNN-GAN	Private	88	EER <4%	Data Aug.	✗	✗
Zhao [12]	2021	Ac, Gy, Or, To	Action	HMM, SVM	Private	104 123	EER <5%	Filtering	✗	✗
MMAuth [13]	2022	To, Ac, Gy, Ma, Or, Loc, App	Touch	NN	Private H-MOG	100 100	EER<15% EER <9%	Multi-modal Wild Dataset	✓	✗
Li [14]	2022	Ac, Gy, Ma	Motion	TF GAN	H-MOG	88	EER <0.1%	Data Aug. Fusin	✗	✗
CT-Auth [15]	2023	To	Touch	LSTM	Private	100	EER <9%	Capacitive Touch	✗	✗
IncreAuth [16]	2023	To, Ac, Gy, Ma, Or	Motion	GBDT NN	Private	100	EER <5%	Context-aware Long Term	✗	✗
AttAuth [17]	2023	To, Ac, Gy, Ma, Or	Touch	RNN LSTM	Private	100	EER <3%	Attention	✗	✗
Hu [18]	2023	Ac, Gy, Ma	Motion	Siamese NN PCA, EE	H-MOG BrainRun	100 100	EER <0.6%	Two Stage	✗	✗
HandPass [19]	2023	Ac	Vibration	Variational Autoencorder	Private	94	EER <3%	Hand Vibration	✓	✗
FingerSlid [20]	2024	Ac, Vi	Vibration	Siamese NN	Private	40	EER <5%	Vibration	✗	✗
MAuGAN [21]	2024	Ac, Gy	Motion	GAN Autoencorder	H-MOG	70	EER <3%	Data Aug.	✓	✗
SNNAuth [22]	2024	Ac, Gy, Ma	Motion	Spiking NN OC-kNN	H-MOG		EER <2%	Spike NN	✗	✗
CAuthNet-OC	2024	To, Ac, Gy, Ma	Touch	Siamese NN EE, OC-SVM, IF	H-MOG	100	EER <0.6%	Touch Avg	✗	✗
CAuthNet				Siamese NN			EER <3%	Portable Privacy Touch Avg	✓	✓

touchscreen. An RNN captures temporal patterns among touch gestures. CT-Auth was validated using the data collected from 100 volunteers over two months. IncreAuth addresses the challenges of characterizing touch operations under complex contexts and maintaining authentication accuracy over time. It uses an incremental learning-based continuous authentication framework for stable performance over long-term usage scenarios [16].

A novel method combining manual construction and deep metric learning for two-stage feature extraction is proposed in [18]. Initially, time-series raw data from accelerometer, gyroscope, and magnetometer sensors is transformed into 69 statistical features. These features are fused into a three-channel matrix and processed by a CNN-based Siesemic network to generate features for classification. The elliptic envelope classifier is used to distinguish users as legitimate or impostors. Despite using a triplet network to generate embeddings for unseen users, final classification relies on multi-class supervised learning, limiting portability.

FingerSlid leverages the unique influence of sliding fingers on active vibration signals for continuous authentication. This system uses vibration motors and accelerometers in mobile devices to sense biometric features of sliding fingers, achieving behavior-independent continuous authentication. Two different vibration signals, the chirp signal and the stable signal, extract distinct biometric features, processed using a Triplet network. A unique Characteristic Behavior Fingerprint (CBFs) is generated for each user during registration. Authentication calculates distances between the current user's feature vector and stored CBFs, identifying the user based on the minimum distance [20].

2.4 Anomaly Detection

In this section, we review anomaly detection techniques from the literature that necessitate user-specific training data for continuous authentication models. The H-MOG approach [7], [23] leveraged multimodal sensors, specifically touch and motion, from 100 volunteers to develop a continuous authentication system. This system achieved an EER of less than 9% by introducing hand movement, orientation, and grasp (HMOG) features. These features unobtrusively capture subtle micro-movements and orientation dynamics from how users grasp, hold, and tap on their smartphones. HMOG features include resistance features, measuring phone micro-movements in response to tap gestures, and stability features, assessing how quickly movement and orientation perturbations dissipate after taps. The system requires 120 seconds of data for effective authentication.

The work in [8] explored a HMM-based classifier to model the distribution density of sensor data sequences for authentication. They employed a one-class learning method to build the authentication model, trained solely on samples from the legitimate user. This approach was tested under various scenarios, showing better performance with longer observation periods or shorter timespans between training and detection phases. This approach obtained EER less than 6% on H-MOG dataset.

MMAuth, described in [13], integrates heterogeneous information from multiple modalities (e.g., motion movement pattern, touch dynamics, usage context). A time-extended behavioral feature set (TEB) and a deep learning-based one-class classifier (DeSVDD) were developed for more accurate authentication. Evaluations on a new, unconstrained smartphone usage dataset collected from 100 volunteers and a publicly available laboratory dataset yielded EER of less than 15% and 9%, respectively. HandPass, a continuous user authentication mechanism, utilizes vibration responses from hand biometrics, passively activated by natural user-device interactions. These responses are embedded in the mechanical vibrations of a force-bearing body, captured by the device's built-in accelerometer [19]. The work [19] exploited autoencoders to achieve EER of less than 3%.

MAuGAN [21] , leveraging both autoencoders and Generative Adversarial Networks (GANs), achieves a EER of less than 3% for continuous authentication. This system adopts a distributed architecture, with the smartphone tasked with real-time authentication using a well-trained autoencoder. To minimize on-device computation, a remote server handles the training of both the Conditional Transformer GANs and the autoencoder. This server-side training also enables MAuGAN to efficiently generate high-quality augmented training data. The core authentication process relies on a memory-augmented autoencoder. This model, trained on legitimate user data, reconstructs incoming user data and calculates the reconstruction error. This error metric is then used for classification, simplifying user identification. Evaluations on data from 70 subjects demonstrate MAuGAN's effectiveness, achieving an EER of less than 3%.

2.5 Improtance of Privacy and Portability

We use two key characteristics, privacy and portability, to distinguish our work, CAuthNet, from these state-of-the-art methods. Therefore, let us explain these two characteristics in detail. The schemes based on classification build classifiers to distinguish legitimate users from imposters [6], [9]–[12], [14]–[20], [22]. This approach requires the collection of sensor data from multiple users by a server to train a user-specific model. For each user, a separate classifier is trained to distinguish between the legitimate user (positive class) and all other users (negative class). Various machine learning algorithms, ranging from SVM to NN, have been used for this purpose. These classifiers learn the patterns and characteristics that differentiate the legitimate user's data from others.

While this approach achieves low EER, it presents significant challenges related to privacy, scalability and portability. Collecting and storing behavioural biometric data from numerous users in a centralised server raises serious privacy concerns. Moreover, the scalability of this method is limited: when a new user is added, all classifiers must be retrained using the new user's sensor data. This retraining process is impractical as it requires each user to generate extensive amounts of sensor data, creating a significant barrier to scalability. Moreover, every user needs their own classifier for continuous authentication, the classification based approaches are not portable.

On the other hand, the anomaly schemes [7], [8], [13], [19], [21] develop user-specific models without needing data from other users, thus preserving data privacy. However, these methods heavily rely on data augmentation techniques, using GAN and transformers to generate large amounts of synthetic data. This means that each user must produce a substantial amount of training data—thousands of touches or hundreds of minutes of motion data—to successfully learn individual user representations. Therefore, anomaly detection methods are not scalable and portable.

2.6 Novelty of the proposed work

In contrast, our proposed work requires data from only a small set of users for training the model. The trained model can then be deployed for new users without the need for additional data collection or training, ensuring privacy, portability and scalability. Therefore, our approach addresses the privacy concerns and practical limitations of previous methods, making CAuthNet a more viable solution for continuous authentication.

The experimental section below showcases the proposed algorithm's remarkable ability to achieve an Equal Error Rate (EER) below 3% when evaluated on user data it has never encountered before, without any need for retraining. To enable continuous authentication of new users, the trained model first extracts a user-specific feature during enrollment. Subsequently, throughout the user session, the model continuously extracts this same feature and compares its similarity to the enrolled feature vector. As long as this similarity surpasses a predefined threshold (established during model training), the user is recognized as genuine. This approach stands in stark contrast to previous works like [18], [20] that utilize seismic networks for feature extraction. These methods require training classification models using the extracted features, a step entirely bypassed by our efficient algorithm.

One of the reasons our work doesn't require training a classifier using the embeddings for continuous authentication is that we meticulously pre-processed the sensor readings to capture high-fidelity touch events. However, we also recognized that even after pre-processing, the touch events can be inherently noisy. To address this, we continuously averaged the touch events before feeding them into the trained model. This process acts as a temporal smoothing filter, effectively reducing noise and enhancing the signal-to- noise ratio of the data. Consequently, this has resulted in generating embeddings which are more robust, representative, and generalizable. This robustness empowers the model to perform accurate continuous authentication solely based on these embeddings. For comparison, as shown as CAuthNet-OC in Table 1, when we train a classifier using the embeddings from the unseen users, the EER is around 0.1%.

CAuthNet takes advantage of the natural way we interact with our phones – through multiple touches. A 2016 Dscout study revealed that users average a staggering 2,617 daily touches, with heavy users reaching an impressive 5,427 [28]. The same study found an average of 145 minutes spent on phones across 76 sessions. Mobile banking transactions often require more than 20 touches, highlighting the practicality of CAuthNet's approach. CAuthNet is the first known approach that develops a portable continuous authentication model to achieve exceptional performance. Even with just 10 touches, CAuthNet delivers a low EER of less than 8%. This EER progressively improves as the number of touches increases, reaching under 5% with 15 touches and a remarkable result of less than 3% EER with 20 touches.

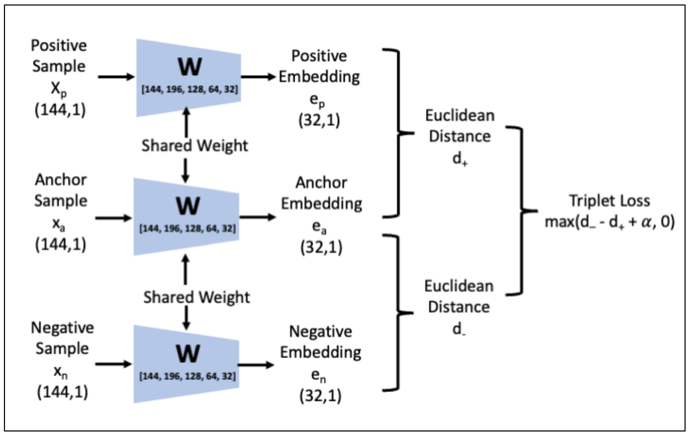


Figure 2: Architecture of Seismic Network.

3 CAuthNet: The Proposed Algorithm

In the context of continuous authentication, the user u's behavioural biometric vector at time t can be defined as a vector denoted by $\mathbf{x}_u^t$. These vectors contain sensor readings to represent the user's actions and patterns. Initially, as shown in Figure 1, the users need to enrol their behavioural biometric vector. During enrollment, users might be asked to perform a number of touch events on their smartphones. The enrolled biometric vectors will be securely stored on the devices for future authentication. During the authentication process, the current behavioural biometric vectors will be compared with the enrolled features. To perform continuous authentication, the similarity between the user behaviour vector at time step t and the enrolled features will be assessed by calculating the distance between the vectors using a suitable distance metric, such as Euclidean distance.

If the distance between the vectors at time t falls below a predefined threshold, indicating that the user's behaviour has remained relatively consistent, the authentication system may continue to validate the user's identity without requiring additional verification steps. However, if the distance exceeds the threshold, suggesting a significant deviation or anomaly in user behaviour, the system may prompt for further authentication measures to verify the user's identity and ensure security.

To achieve accurate similarity measurements, a robust machine-learning model is required. This model should extract informative features (embeddings) from the user's behavioural biometric vector. Ideally, these embeddings should exhibit a property called clustering: embeddings from the same user should be closer together in the embedding space compared to those from different users. This clustering property enables the distance metric to distinguish between legitimate and fraudulent user behaviour effectively. To develop such a robust model, we use a Seismic network-based one-shot learning algorithm [25].

The concept of Seismic networks, popular in image recognition tasks such as face verification and signature verification [25]–[27], can be extended to tasks like continuous authentication. As shown in Figure 2, Siamese networks operate by comparing inputs and determining their similarity or dissimilarity. In this approach, the inputs are passed through a neural network one by one, and the resulting outputs are compared to assess their similarity. To achieve this, a neural network is trained such that if the inputs are from the same user, the generated outputs (embeddings) will be close to each other in the feature space. Conversely, if the inputs are from different users, the embeddings will be distant from each other. This is achieved by using a triplet loss function during training, which penalizes the network for producing embeddings that are too far apart for similar pairs and too close for dissimilar pairs. As a result, the network learns to produce similar embeddings for behavioural biometric vectors from the same user and distinct embeddings for those from different users.

Let us demonstrate how a one-shot Seismic architecture can be adapted for continuous authentication. Let's begin by defining a set of users $u \in \{1, \ldots, U\}$, where U represents the total number of users selected for training the model. Furthermore, let's denote the user u's sensor data at time t as $\mathbf{x}_u^t$ and the machine-learning model to be trained on using the data as $\mathbf{W}$. The aim of $\mathbf{W}$ is to extract embedding vector $\mathbf{e}_t^u$ corresponding the the data $\mathbf{x}_t^u$. Let's define this using the following equation:

$$\mathbf{e}_t^u = \mathbf{W}(\mathbf{x}_t^u), \forall u, t. \tag{1}$$

Let us suppose that a server collects data from users who are selected for the training. The server randomly initializes $\mathbf{W}$ and extracts the corresponding embeddings for all u and t. Then the server computes the loss using triplet loss. The triplet loss is calculated by comparing pairs of embeddings and enforcing similarity for pairs of the same user while encouraging dissimilarity for pairs of different users. The server needs to set up a training dataset to perform the triplet loss-based model training.

3.1 Setup Training Dataset

To configure the training dataset, the server categorizes embeddings into triplet sets, comprising anchor, positive, and negative embeddings. Anchor and positive embeddings belong to the same user, capturing behavioural patterns observed at different time points. These triplets are instrumental in training the model to recognize and reinforce the consistency of a user's behaviour over time. Conversely, negative embeddings are paired with anchor embeddings from different users, representing instances where behavioural and contextual patterns diverge significantly. These triplets

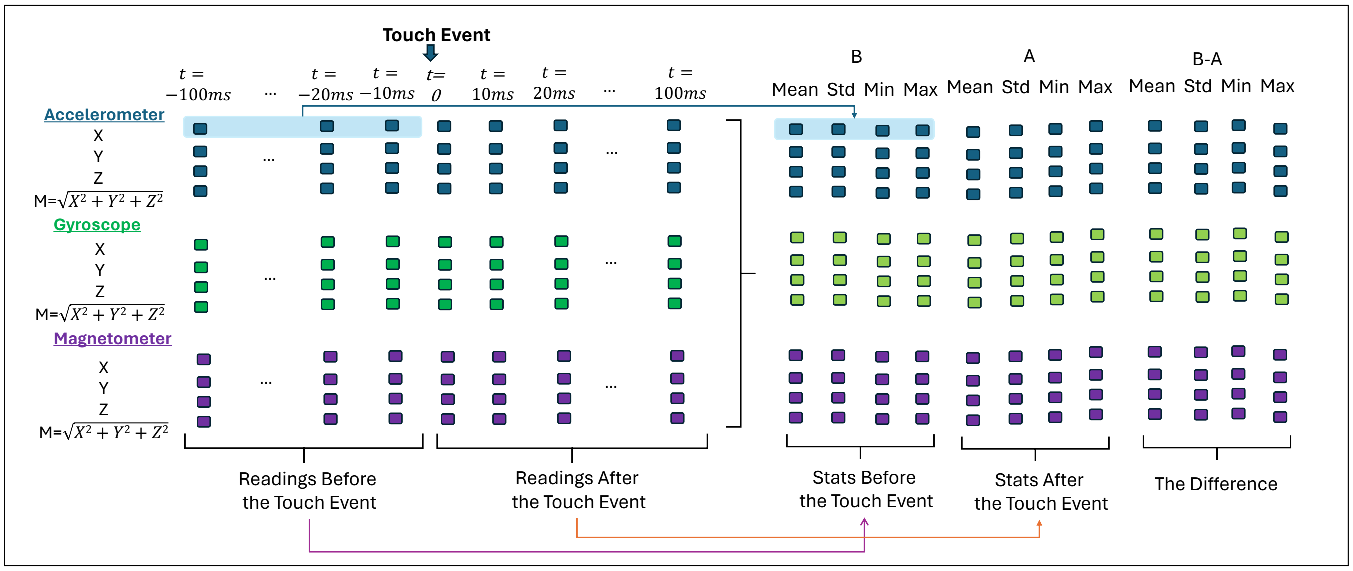


Figure 3: Block diagram illustrating the extraction of 144 features from all three sensors for each touch event.

facilitate the model in learning to distinguish between genuine users and potential adversaries by emphasizing differences in their behavioural signatures.

To determine the composition of the training dataset, we calculate the number of triplets formed from the embeddings generated by U users over T time points. Each user contributes combinations of embeddings from T time points, resulting in a total of $U \times \binom{T}{2} = U \times \frac{T(T-1)}{2}$ anchor-positive pairs. While there are a significantly large number of anchor-negative pairs, the server selects only $U \times \binom{T}{2} = U \times \frac{T(T-1)}{2}$ number of hard negative points to match the anchor-positive pairs. Therefore, there will be $U \times \binom{T}{2} = U \times \frac{T(T-1)}{2}$ number of triplets in the training dataset. Out of $U \times \binom{T}{2} = U \times \frac{T(T-1)}{2}$ triplet pairs, only a subset of triplet pairs will be used for training.

3.2 Triplet loss

The triplet loss function, as shown in Figure 2, requires three embeddings *anchor:* $\mathbf{e}_a$, *positive:* $\mathbf{e}_p$, and *negative:* $\mathbf{e}_n$. The anchor and positive embeddings are from the same user while the negative embedding is from another user. First, the distance between the anchor-positive pair, denoted as d_+, as well as the distance between the anchor-negative pair, denoted as d_-, is calculated using Euclidean distances. The loss is the difference between d_+ and d_-. If the distance between the anchor and the positive sample is smaller than the distance between the anchor and the negative sample by more than the margin α, the loss is zero, indicating that the triplet satisfies the desired condition. However, if the distance difference falls below the margin, the loss becomes positive, prompting the model to adjust the weights to increase the separation between the positive and negative pairs. By optimizing the triplet loss during training, the Siamese network learns to produce embeddings that effectively capture the desired relationships between samples, enabling accurate discrimination between data points from owners and adversaries.

The triplet loss function plays a crucial role in training by ensuring that the distance between the embeddings of anchor-positive pairs is less than that between anchor-negative pairs by at least a predefined margin α. This objective drives the machine learning model to produce embeddings that accurately reflect the inherent similarities and differences between input pairs, thereby enhancing its ability to differentiate between data from legitimate users and adversaries. During training, the server computes the triplet loss across all pairs in the dataset, leveraging this loss to update the model weights via backpropagation. Successfully training a Siamese network with triplet loss necessitates meticulous preprocessing of sensor data and the extraction of robust features to ensure the model captures meaningful patterns and relationships. This careful data preparation is essential to achieve high accuracy in distinguishing genuine user interactions from adversarial attempts.

4 Data Processing and Feature Extraction

In this section, we delve into the data processing and feature extraction techniques employed to prepare the sensor data for training the Siamese network. The raw sensor data undergoes several preprocessing steps, including normalization, noise reduction, and segmentation, to ensure consistency and relevance. We then extract robust features that capture the users' behavioural patterns in the data. These features serve as the foundation for the model's ability to differentiate between genuine user behaviour and adversarial actions, facilitating the accurate generation of embeddings required for effective triplet loss optimization.

4.1 Dataset

We use H-MOG dataset [23] to validate the proposed algorithm.The H-MOG dataset contains data collected from 100 smartphone users (53 male, 47 female) and it comprises accelerometer, gyroscope, and magnetometer readings when users interacted with smartphones via touches, scrolls, and

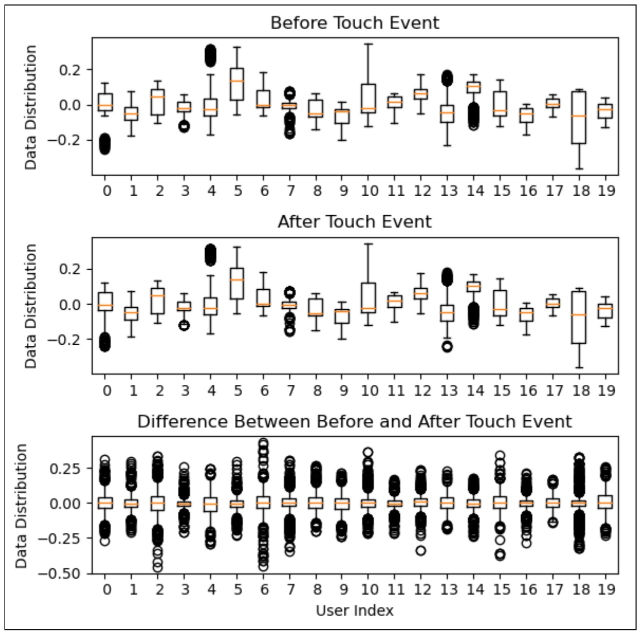


Figure 4: Distribution of Feature 1 - accelerometer X-axis mean readings for 20 users.

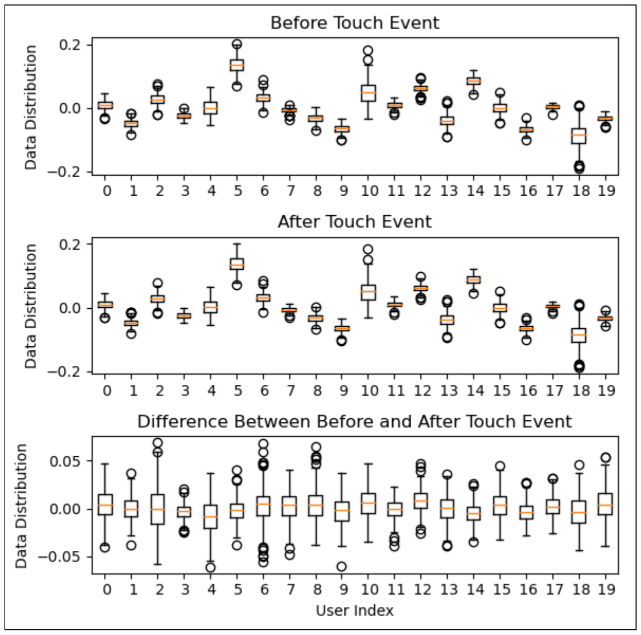


Figure 5: Distribution of Feature 1 - accelerometer X-axis mean readings for 20 users after averaging 20 touch events.

pinches. Each user data collected from 24 sessions included various combinations of activities such as document reading, text production, or map navigation while sitting or walking where each activity lasted approximately 5 to 15 minutes, totalling approximately 2 to 6 hours of sensor data per user. These data were recorded at a 100Hz sampling rate. It should be noted that H-MOG is the only publically available dataset that contains both continuous motion sensor data and touch sensor data. [1] We used half of the users' data to train the CAuthNet model $\mathbf{W}$, and we tested the model using the data from the remaining users.

4.2 Data Processing

We utilised sensor readings associated with touch events. Some sessions are shorter and some sessions are extremely longer. For each user, we utilised data from 1200 touch events, totalling around $120,000$ touch events. We utilized sensor readings captured during two specific time intervals surrounding each touch event. As shown in Figure 3, the first interval spans from 100 milliseconds before the touch event to the moment of the touch event, while the second interval extends from the touch event time to 100 milliseconds after it. Given that the data was sampled at 100Hz, we utilised 20 readings from each sensor for every touch event.

4.3 Feature Selection

Each sensor records data along the X, Y, and Z axes. Additionally, as recommended in [24], we've augmented another reading, M, computed as $\sqrt{X^2+Y^2+Z^2}$. Consequently, we now have 12 features derived from the readings of all three sensors i.e., $3\times$ {X, Y, Z, M}. Given that we use 10 readings before and 10 readings after each touch event, we can calculate the maximum, minimum, mean, and standard deviation for each of the 12 features before and after the event. Subsequently, we can determine the difference between these calculated values before and after the touch event. In total, this results in 12 features multiplied by 4 statistics (maximum, minimum, mean, standard deviation) calculated for each of the 3-time points (before, after, and their difference), leading to a total of 144 features per touch event (see Figure 3 for more details).

Fig. 4 shows the distribution of feature 1 (out of 144) which represents the mean of X-axis accelerometer readings for twenty users. The top plot in Fig. 4 represents the distribution before the touch event, the middle plot is after the touch event and the bottom one represents the difference between before and after touch events. At a high level, this figure demonstrates that different users have different signatures. However, numerous outliers need to be appropriately handled. We use an average of multiple consecutive touch events to perform temporal smoothing as described below.

Various techniques exist to address outliers in data, including imputation, trimming, and log transformation. However, our work found that simply averaging multiple touch events effectively reduces their influence. As described earlier, each touch event is converted into a 144-feature data point. To achieve this averaging, we calculate the mean of a specified number, denoted as N_{avg}, of consecutive touch events. We denote the 144 feature extracted from individual touch events as (TE_i). Let's denote the averaged touch events as (ATE_i) where (ATE_i) can be calculated as follows:

$$\text{ATE}_i = \frac{1}{N_{\text{avg}}} \sum_{j=1}^{N_{\text{avg}}} \text{TE}_{i+j}.$$

Fig. 5 illustrates the data distribution after 20 touch events are averaged. From Fig. 4 and Fig. 5, it is clear that averaging the touch events smooths out the data distribution

1. The BrainRun dataset [18] does not provide continuous motion sensor reading before and after the touch event.

by minimizing the effects of outliers. Note that, for a given touch event, we select several touch events that are adjacent and then calculate the mean of those touch events. In this work, we consider multiple scenarios, averaging from 2 to 20 touch events. Now we can use these extracted features to train the Seisemic architecture.

4.4 Seisemic Network Architecture and Model Training

We trained several models using various neural network architectures. The architecture comprises a series of fully connected layers. It begins with an input layer of size 144, followed by several hidden layers with neurons activated by ReLU. The output layer reduces the dimensionality to embeddings with dimension 32, followed by another ReLU activation and a batch normalization layer for normalization. During the forward pass, input data is passed through the fully connected layers, and the output is normalized to have a unit norm along the embedding dimension. This normalization ensures consistency in the scale of the embeddings produced.

The architecture accepts three input samples ('anchor', 'positive', and 'negative') and independently processes each through the same set of layers defined earlier. The embeddings for these inputs are then used to compute the Triplet loss. For training, we randomly select the data from 50 users to train the CAuthNet model. We constructed five distinct models, gradually increasing the number of users needed for training per model: 10, 20, 30, 40, and finally 50 users. We organized the data into 128 batches. Each batch comprised 10 triplets per user, resulting in 500 triplets per batch for the fifth model. Following a recommended procedure [25], for each triplet, we randomly designated the anchor and positive pairs. However, for the negative pair, we employed a method where we randomly selected 16 negative samples and then chose the one closest (most similar) to the given anchor data point. This selection of a similar negative point was facilitated by utilizing the latest updated model trained on the last batch.

4.5 Impact of Number of Users for Training

First, we examine how model accuracy varies with the number of users involved in training. To do this, we train five different models, each with a varying number of users in the training dataset. The model weights are initialized using He initialization, and biases are set to zero. As mentioned earlier, each user contributed 1200 touch events. For this experiment, we randomly select 300 ATEs per user, each derived by averaging adjacent 20 touch events.

Table 2 outlines the hyperparameters used for the five models. The first model, M_{10}, was trained with data from 10 users. We created 64 batches for training, each batch containing 10 triplets per user, totaling 100 triplets per batch. A Siamese network with one hidden layer was sufficient to achieve a model with minimal loss. Similarly, four additional models were trained with an increasing number of users. The last model M_{50} in Table 2 was trained with data from 50 users, utilizing a Siamese network with 4 hidden layers. For all models, the input layer processes 144-dimensional data features, and the output layer, with 32 neurons, corresponds to the number of embeddings.

The training loss is around 0.1 for all five models. In Figure 6, the first column displays the training loss trajectories for models M_{10}, M_{30}, and M_{50}, demonstrating stable convergence. The middle column illustrates the distribution of Euclidean distances between embeddings of the same users (depicted in blue) and different users (depicted in pink). This separation in distances evidences the models' ability to effectively differentiate between genuine and adversarial user interactions without overfitting. Analysis of these distance distributions suggests that setting a threshold of approximately 1 during testing can effectively discriminate between genuine users and adversaries. This is also observed in Figure 7.

The second part of Table 2 presents the test results for the trained models. To evaluate these models, we utilized data from 50 users who were not included in the training set. For each user, 300 ATEs were randomly selected, leading to a total of 15,000 ATEs. Each ATE was generated by averaging 20 consecutive touch events. Using the trained models, we extracted embeddings for all 15,000 ATEs. Using the 15,000 embeddings obtained from the test dataset, we randomly generated 15,000 similar pairs (where both embeddings belong to the same user) and 15,000 dissimilar pairs (where the embeddings belong to different users). The third column in Figure 6 shows the Euclidean distance distributions of these pairs. This figure illustrates that, as the number of users in the training dataset increases, the overlap between the distances of the same-user pairs (intra-user distances) and different-user pairs (inter-user distances) decreases. This trend signifies an improvement in the model's ability to distinguish between genuine and malicious users with increased training diversity, thereby enhancing its overall performance.

To evaluate the performance of the models, we calculate the True Positive Rate (TPR) and True Negative Rate (TNR). TPR measures the proportion of actual positives that are correctly identified. In this context, it indicates the percentage of touch events from the same user that are correctly identified as such. We calculate TPR using TPR $= \frac{\text{TP}}{\text{TP+FN}} \times 100$, where TP (True Positive) indicates the pairs of touch events from the same user that are correctly identified as being from the same user and FN (False Negatives) indicates the pairs of touch events from the same user that are incorrectly identified as being from different users.

TNR measures the proportion of actual negatives that are correctly identified. Here, it indicates the percentage of touch events from different users that are correctly identified as such. We calculate TNR using TNR $= \frac{\text{TN}}{\text{TN+FP}} \times 100$, where TN (True Negatives) indicates the pairs of touch events from different users that are correctly identified as being from different users and FP (False Positives) indicates the pairs of touch events from different users that are incorrectly identified as being from the same user.

For each pair of embeddings, we compute the Euclidean distance. We use a threshold distance to classify the Euclidean distances. If the pair's Euclidean distance is below this threshold then those paid is classified as being from the same user (positive), and distances above are classified as being from different users (negative). Table 2 shows the TPR and TNR for all five models at the optimal operating point where TPR=TNR. Increasing the number of users during the

Table 2: The hyperparameters used for training the model in the centralised scenario.

Training						Testing			
Model Name	# of Users	# of Batches	# of Triplets / Batch	Network Architecture	Training Loss	# of Users / Testing	# of Pairs	TPR	TNR
M_{10}	10	128	100	[144,64,32]	<0.08	50	225M	86%	86%
M_{20}	20	128	200	[144,64,32]	<0.11	50	225M	89%	89%
M_{30}	30	256	300	[144,196,128,64,32]	<0.04	50	225M	92%	92%
M_{40}	40	256	400	[144,196,128,64,32]	<0.06	50	225M	94%	94%
M_{50}	50	256	500	[144,196,128,64,32]	<0.08	50	225M	97%	97%

training results in increasing accuracy increasing from 86% to 97%. Therefore, we would use all fifty users for training a model in the following experiments.

Figure 7 illustrates the False Positive Rate (FPR), TPR, and the average of TPR and True Negative Rate (TNR) for various thresholds ranging from 0 to 1.8. The optimal threshold is identified at approximately 1, where the TPR and TNR converge, yielding a TPR and TNR both at 97%, which equates to an EER of 3%. This threshold effectively balances the model's accuracy in distinguishing between genuine users and adversaries, maximizing detection performance while minimizing both false positives and false negatives.

4.6 Impact of Averaging the Touch Events

A significant finding of this research is the beneficial impact of averaging touch events on both training and testing the models. As depicted in Figures 4 and 5, averaging touch events substantially mitigates the influence of outliers, leading to more stable and reliable data. To quantify this impact, we conducted experiments by varying the number of touch events averaged, ranging from 2 to 20, while maintaining the same hyperparameters used for model M_{50} as detailed in Table 2. The only modification was the number of touch events per Aggregated Touch Event (ATE). Figure 8 showcases the accuracy, defined as the point where TPR equals TNR, for different values of N_{avg}. The proposed method achieved an accuracy of 77% with only two touch events, improved to 84% with four touch events, reached 93% with ten touch events, and attained up to 97% accuracy when N_{avg} was set to 20. This demonstrates a clear trend: increasing the number of averaged touch events significantly enhances user detection accuracy.

To further illustrate the effectiveness of averaging touch events, we employed t-distributed Stochastic Neighbor Embedding (t-SNE) to visualize the high-dimensional embeddings. t-SNE is a dimensionality reduction technique that maps high-dimensional data to two or three dimensions while preserving the relative distances and structure of the data. This visualization helps to observe the clustering behavior of the embeddings. The t-SNE plots in Figure 10 show the embeddings for all 50 unseen users with different N_{avg} values, where the dimensions are reduced from 32 to 2. Different colors represent different users, forming distinct clusters. As the number of ATEs increases, the clusters become more defined and separated, indicating improved model performance in distinguishing between users.

5 Comparision with other works

Table 1 in Section 2 compares the proposed approach with 17 other related works. Unlike these studies, our approach supports the portability property, allowing it to generalize across different users without the need for retraining. While a direct comparison is challenging due to this unique feature, our model achieves a 3% EER, which is competitive with the accuracies reported by other works. To further demonstrate the versatility and effectiveness of the embeddings generated by the CAuthNet model, we have developed a variant: CAuthNet-OC (One Class) and we used this variant to compare the existing works.

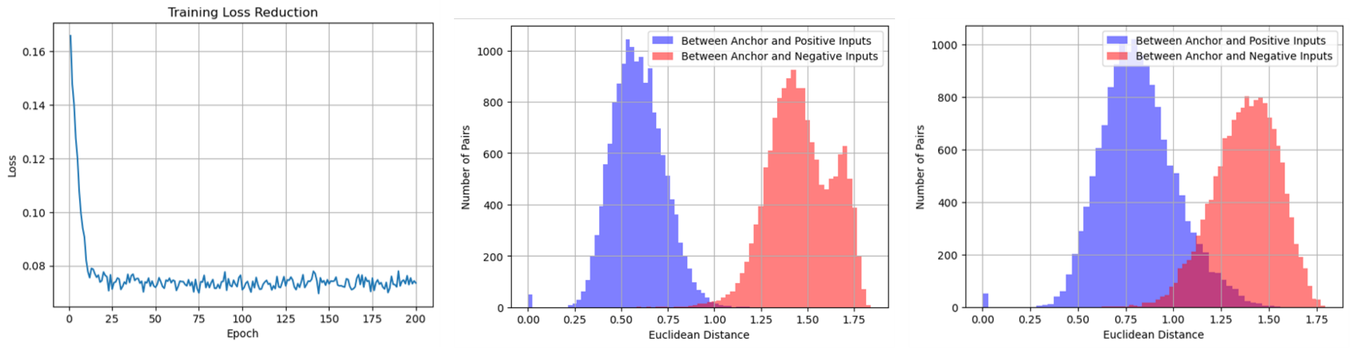
5.1 CAuthNet-OC

In the CAuthNet-OC variant, we utilize the embeddings generated by the CAuthNet model to train a one-class model for each user. This approach leverages high-dimensional embeddings to capture the distinctive patterns of a user's touch interactions. A one-class classification model is trained to recognize these patterns. During testing, the model's decision function outputs a confidence score indicating whether the input belongs to the user. This approach allows for efficient user identification and leverages the discriminative power of the CAuthNet embeddings to achieve accurate and reliable user authentication.

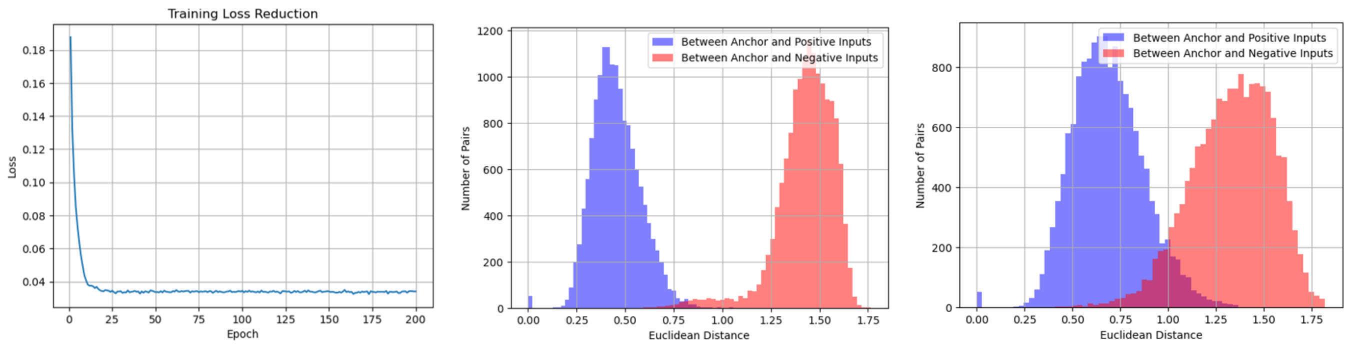
Table 3: Comparing the EERs of the CAuthNet-OC (variant of the proposed methods) and Two-stage fusion method of [18].

	Seen Users (EER)		Unseen Users (EER)	
	[18]	CAuthNet-OC	[18]	CAuthNet-OC
EE	0.56%	0.53%	-	0.54%
OC-SVM	1.73%	1.14%	-	1.17%
IF	9.75%	1.09%	-	1.12%

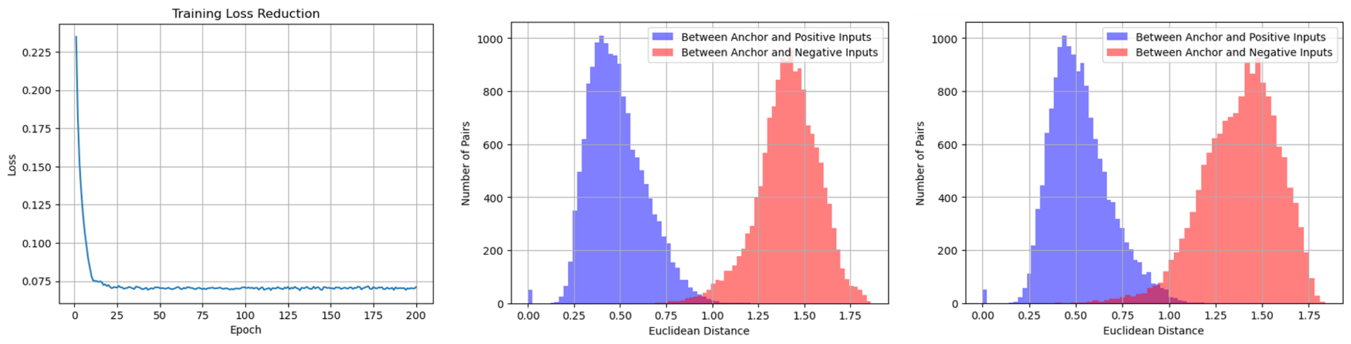
As per Table 1, the two-stage fusion approach in [18] reported a low EER of 0.56% compared to other works that use a one-class classifier. Therefore, we focus on [18] to perform this comparison. In [18], the authors propose to use CNN-based seismic network to generate 512-dimensional embeddings followed by one-class classification using algorithms such as One-Class Support Vector Machine (OC-SVM), Isolation Forest (IF), and Elliptic Envelope (EE). The work in [18] also used the H-MOG dataset for validation. In [18], embeddings for all the users were obtained using the CNN-based seismic network, hence all the users were seen by the trained seisemic model. However, in our work, we use only 50 users for training the model and we kept the other 50 users (unseen users) only for testing. Table 3 shows the EERs for both the method for both seen and



When ten users are used for training the model.



When thirty users are used for training the model.



When fifty users are used for training the model.

Figure 6: The first column shows the training loss; the second column shows the Euclidean distances between the embeddings of the same users (blue) and different users (pink) who were part of the training; and the third column shows the Euclidean distances between the embeddings of the same users (blue) and different users (pink) who were NOT part of the training.

unseen users. Table 3 presents the EERs for both methods on seen and unseen users. The results from the table clearly demonstrate the superiority of CAuthNet-OC. For all three one-class classification algorithms (EE, OC-SVM, and IF), CAuthNet-OC achieves lower EERs on both seen and unseen users compared to the two-stage fusion method. Refer to Subsection 6.3 for complexity comparison between CAuthNet and the model in [18]. Fig. 9 compares EERs of the proposed methods and the state-of-the-art schemes.

5.2 Comparision with Data Augmentation Approaches

This section presents a comparative analysis between the proposed CAuthNet and data augmentation-based anomaly detection techniques. According to the literature review summarized in Table 1, the approach described in [14] achieves an exceptionally low EER of less than 0.1%, which is significantly better than other data augmentation methods.

In [14], transformer-based GANs were employed to augment user sensor data, subsequently used to train a CNN model for feature extraction. The deep features extracted by the CNN model were then applied for one-class classification using SVM. However, implementing this method in practical applications poses significant challenges. Specifically, the model uses three transformers — one each for the accelerometer, gyroscope, and magnetometer. Each transformer has approximately 0.35 million parameters and requires around 4.5 million multiplications per epoch. On top of these, there are two more CNN models.

Training these transformer-based GANs is particularly

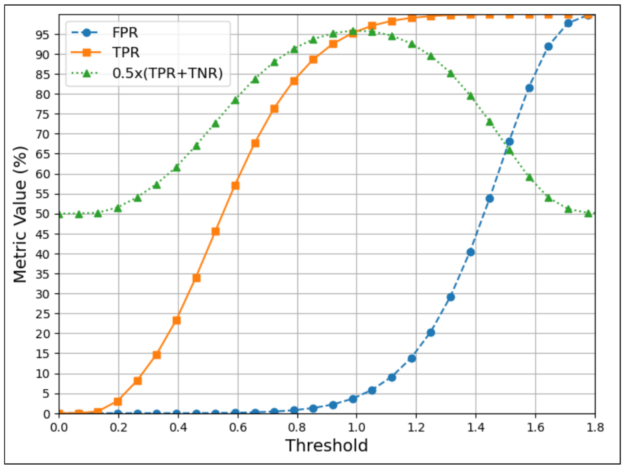


Figure 7: Accuracies for the model when $N_{avg} = 20$.

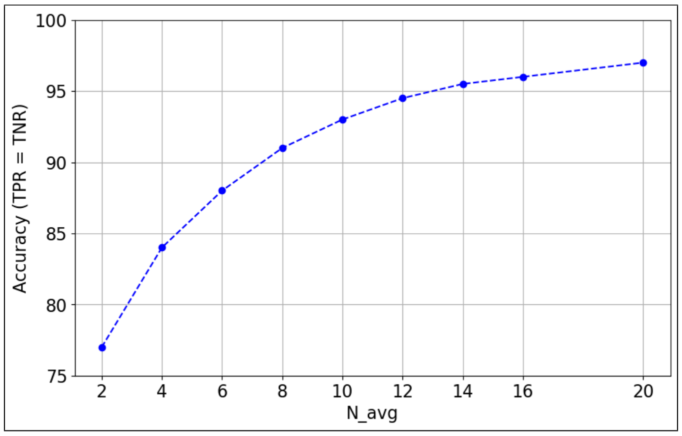


Figure 8: Accuracy increasing with the number of touch events averaged.

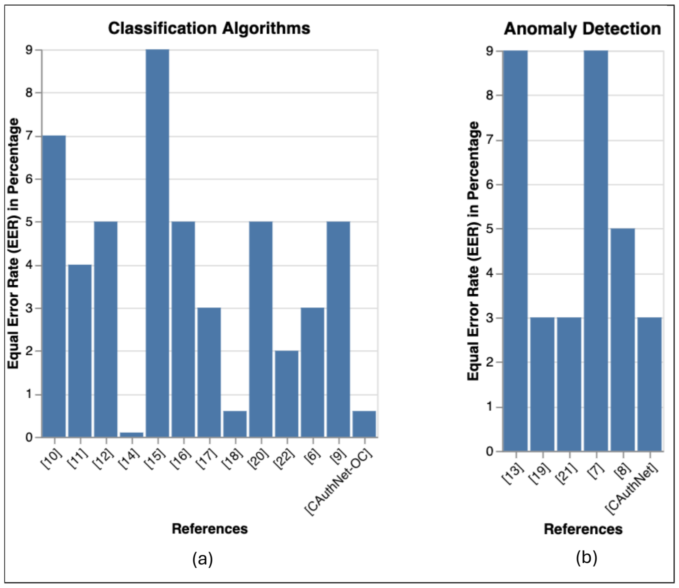


Figure 9: Accuracy comparison of the proposed algorithms against the state-of-the-art schemes in terms of EER. Fig. (a) compares classification based algorithms and Fig. (b) compares anomaly detection based algorithms.

resource-intensive. Each user must generate about 100 minutes of sensor data and train the model for nearly 2000 epochs. This high computational demand places a significant burden on smartphones, making on-device training impractical. As a result, users would likely need to outsource computations to a third-party server, raising concerns about potential privacy violations. It should also be highlighted that this approach is not portable. Consequently, every user must undergo the entire data generation and training process during the enrollment phase to successfully continue authentication.

6 Security, Privacy and Complexity Analysis

6.1 Security Analysis

As demonstrated in the experimental section, the proposed CAuthNet achieves 97% TPR and TNR rates in distinguishing between genuine users and adversaries when an average of 20 touch events are analyzed. However, it is important to note that 20 touch events are typically insufficient to complete any meaningful tasks on smartphones. To assess the robustness of the CAuthNet model, we conducted an experiment where we interspersed touch events from adversarial users with those from genuine users.

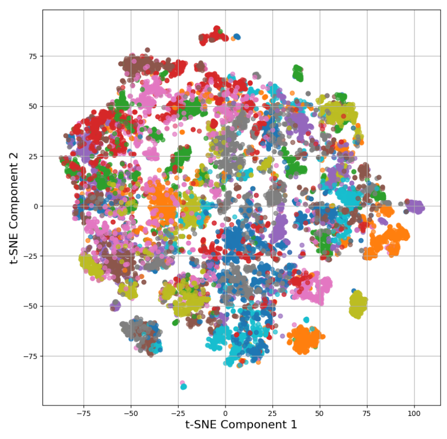
Specifically, we investigated scenarios where an adversary might hijack a smartphone after a few touch events initiated by the genuine user. For instance, we examined the likelihood of detecting such a hijack if it occurs after just two touch events by the legitimate user. The results, depicted in Figure 11, show the detection probabilities based on the number of subsequent adversarial touch events. The CAuthNet model demonstrated a detection probability exceeding 50% after 10 adversarial touch events, rising to over 65% after 12 touch events, and surpassing 75% after 14 touch events.

These results highlight the effectiveness of CAuthNet in dynamically identifying adversarial activities even in mixed touch event scenarios, thus underscoring its potential for enhancing security on smartphones.

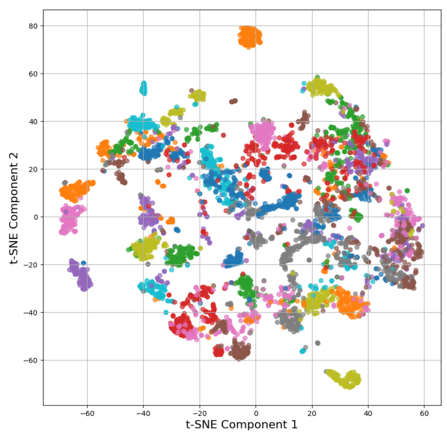
6.2 Privacy Analysis

The CAuthNet model was developed with privacy considerations in mind, leveraging a small dataset from a limited set of users for its training process. This approach ensures that extensive personal data collection is not required, thus safeguarding user privacy. Furthermore, the trained model is designed to be highly adaptable and can be deployed on new users' devices without necessitating additional data collection for re-training. This capability not only enhances the model's practicality and scalability but also maintains user privacy by eliminating the need for continuous data acquisition and storage, thereby minimizing potential privacy risks associated with data collection and handling.

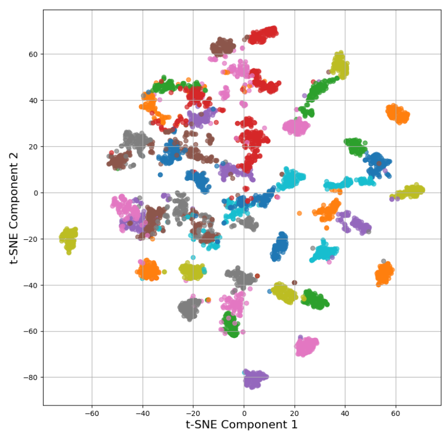
To protect the privacy of the users who shared the data for training the model, we are exploring the federated learning paradigm as a future direction. This paradigm enables on-device training, allowing CAuthNet to learn from diverse user interactions without centralized data storage or transmission. By utilizing federated learning, we aim to



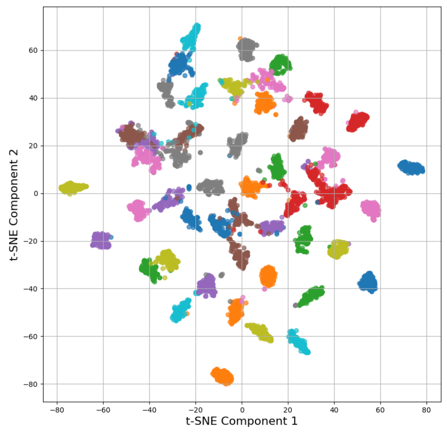
(a) When $N_{\text{avg}} = 2$



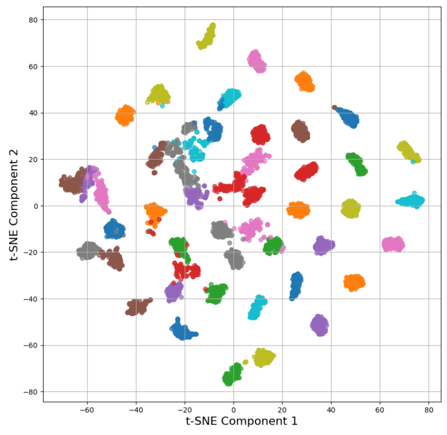
(b) When $N_{\text{avg}} = 4$



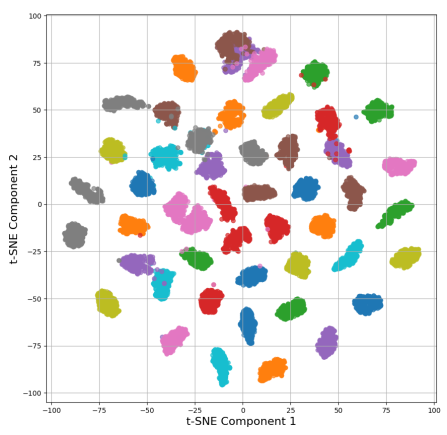
(c) When $N_{\text{avg}} = 8$



(d) When $N_{\text{avg}} = 12$



(e) When $N_{\text{avg}} = 16$



(f) When $N_{\text{avg}} = 20$

Figure 10: tSNE plots for all the 50 unseen users's Embeddings for different values of N_{avg}. The embedding dimensions are reduced from 32 to 2 using t-SNE. Different colours/clusters represent different users.

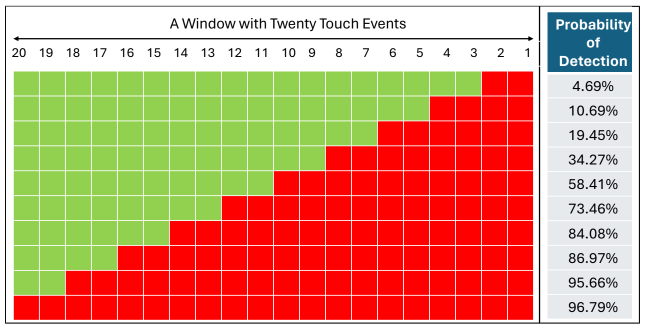


Figure 11: Probability of detecting an attack when an adversary hijacks a smartphone. Green and red blocks denote the touch events by genuine users and adversaries, respectively.

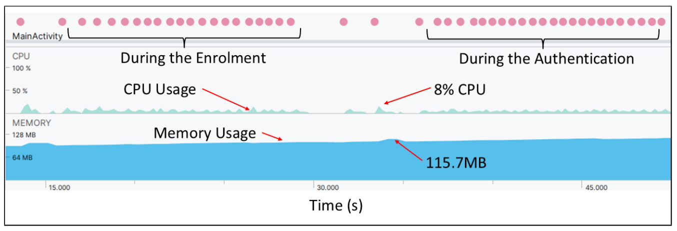


Figure 12: CPU and memory usage profiling of CAuthNet on Android Google Nexus phone.

create a more robust authentication system that protects user data throughout both the training and deployment phases, ensuring that even training data remains secure, as only model updates, not the raw data itself, are shared with the central server.

6.3 Complexity Analysis

The CAuthNet model processes 144-dimensional feature vectors to produce 32-dimensional embeddings, incorporating three hidden layers comprising 192, 128, and 64 neurons, respectively (refer to Table 2). This architecture necessitates approximately 238 KB of memory, primarily due to the total weight count of around 60,800. Such a compact memory footprint renders the model well-suited for deployment on a variety of modern smartphones. The computational demand is driven by the forward pass multiplications, totaling approximately 60,416 multiplications per input.

To evaluate the feasibility on a 1 GHz single-core CPU, which operates at 1 billion cycles per second, we assume one multiplication per cycle. Therefore, the number of multiplications per second would be 1 billion, resulting in an execution time of about 60.4 microseconds per input. This calculation demonstrates that even on a modest 1 GHz single-core CPU, the CAuthNet model's computations are executed efficiently within approximately 60.4 microseconds per input. Thus, the model is suitable for deployment on a broad spectrum of devices, including those with limited processing capabilities, without imposing significant computational delays. To confirm this, we have implemented an Android app using CAuthNet and tested it using a Google Nexus phone. Fig. 12 shows the CPU and memory usage footprint obtained via Android Studio Profile [2]. The developed Android app consumes a maximum of 115MB of memory and 8% of the CPU.

In contrast, the seismic network architecture described in [18] demonstrates considerably higher complexity. This CNN-based model utilizes three convolutional layers, necessitating around 0.3 million weights, and thus requires approximately 1 MB of memory—more than four times the memory required by CAuthNet. Additionally, the computational complexity of the seismic network is notably higher, involving nearly 12 million multiplications per input as compared to CAuthNet's 60,416 multiplications. Given a 1 GHz single-core CPU capable of one multiplication per cycle, processing a single input through the seismic network would take roughly 12 milliseconds, making it almost 200 times slower than CAuthNet.

7 Conclusion

This paper presents CAuthNet, a novel system for continuous smartphone authentication that achieves high accuracy with a remarkably simple and efficient architecture. CAuthNet leverages a one-shot Seismic network specifically tailored for touch events. This enables it to learn user-specific touch patterns from a limited amount of training data, enhancing privacy and reducing storage requirements. CAuthNet achieves impressive results on unseen users from the H-MOG dataset, a large public dataset containing smartphone sensor data. It delivers a low Equal Error Rate (EER) of 3%. This performance is achieved with a lightweight architecture that demands minimal computational resources. This makes CAuthNet highly suitable for real-world deployment on smartphones. Furthermore, exploiting the embeddings generated by CAuthNet within one-class anomaly detection algorithms yields even more promising results. This approach achieves an exceptionally low EER of around 0.5%. Beyond its high accuracy, CAuthNet offers significant advantages compared to existing methods. Unlike approaches such as two-stage fusion and data augmentation, CAuthNet is a more efficient and portable solution. It requires significantly less memory and processing power, making it ideal for resource-constrained mobile devices. Overall, CAuthNet perfectly balances security and efficiency, making it well-suited for real-world implementation.

References

[1] Alzubaidi, A. and Kalita, J., 2016. Authentication of smartphone users using behavioral biometrics. IEEE Communications Surveys & Tutorials, 18(3), pp.1998-2026.

[2] Abuhamad, M., Abusnaina, A., Nyang, D. and Mohaisen, D., 2020. Sensor-based continuous authentication of smartphones' users using behavioral biometrics: A contemporary survey. IEEE Internet of Things Journal, 8(1), pp.65-84.

[3] Chen, F., Xin, J. and Phoha, V.V., 2024. SSPRA: A Robust Approach to Continuous Authentication Amidst Real-World Adversarial Challenges. IEEE Transactions on Biometrics, Behavior, and Identity Science.

[4] Acar, A., Aksu, H., Uluagac, A.S. and Akkaya, K., 2020. A usable and robust continuous authentication framework using wearables. IEEE Transactions on Mobile Computing, 20(6), pp.2140-2153.

[5] Adel, O., Soliman, M. and Gomaa, W., 2021, July. Inertial gait-based person authentication using siamese networks. In 2021 International joint conference on neural networks (IJCNN) (pp. 1-7). IEEE.

[6] Frank, M., Biedert, R., Ma, E., Martinovic, I. and Song, D., 2012. Touchalytics: On the applicability of touchscreen input as a behavioral biometric for continuous authentication. IEEE transactions on information forensics and security, 8(1), pp.136-148.

[7] Sitová, Z., Šeděnka, J., Yang, Q., Peng, G., Zhou, G., Gasti, P. and Balagani, K.S., 2015. HMOG: New behavioral biometric features for continuous authentication of smartphone users. IEEE Transactions on Information Forensics and Security, 11(5), pp.877-892.

[8] Shen, C., Li, Y., Chen, Y., Guan, X. and Maxion, R.A., 2017. Performance analysis of multi-motion sensor behavior for active smartphone authentication. IEEE Transactions on Information Forensics and Security, 13(1), pp.48-62.

[9] Li, Y., Hu, H. and Zhou, G., 2018. Using data augmentation in continuous authentication on smartphones. IEEE Internet of Things Journal, 6(1), pp.628-640.

[10] Abuhamad, M., Abuhmed, T., Mohaisen, D. and Nyang, D., 2020. AUToSen: Deep-learning-based implicit continuous authentication using smartphone sensors. IEEE Internet of Things Journal, 7(6), pp.5008-5020.

[11] Li, Y., Luo, J., Deng, S. and Zhou, G., 2021. CNN-based continuous authentication on smartphones with conditional Wasserstein generative adversarial network. IEEE Internet of Things Journal, 9(7), pp.5447-5460.

[12] Zhao, G., Zhang, P., Shen, Y. and Jiang, X., 2021. Passive user authentication utilizing behavioral biometrics for IIoT systems. IEEE Internet of Things Journal, 9(14), pp.12783-12798.

[13] Shen, Z., Li, S., Zhao, X. and Zou, J., 2022. MMAuth: A continuous authentication framework on smartphones using multiple modalities. IEEE Transactions on Information Forensics and Security, 17, pp.1450-1465.

[14] Li, Y., Liu, L., Qin, H., Deng, S., El-Yacoubi, M.A. and Zhou, G., 2022. Adaptive deep feature fusion for continuous authentication with data augmentation. IEEE Transactions on Mobile Computing.

[15] Shen, Z., Li, S., Zhao, X. and Zou, J., 2023. CT-Auth: Capacitive touchscreen-based continuous authentication on smartphones. IEEE Transactions on Knowledge and Data Engineering.

[16] Shen, Z., Li, S., Zhao, X. and Zou, J., 2023. IncreAuth: Incremental learning based behavioral biometric authentication on smartphones. IEEE Internet of Things Journal.

[17] Zhao, C., Gao, F. and Shen, Z., 2023. AttAuth: An Implicit Authentication Framework for Smartphone Users Using Multi-Modality Data. IEEE Internet of Things Journal.

[18] Hu, M., Zhang, K., You, R. and Tu, B., 2022. Multisensor-Based Continuous Authentication of Smartphone Users With Two-Stage Feature Extraction. IEEE Internet of Things Journal, 10(6), pp.4708-4724.

[19] Cao, H., Jiang, H., Yang, K., Chen, S., Wu, W., Liu, J. and Dustdar, S., 2023. Data Augmentation-Enabled Continuous User Authentication via Passive Vibration Response. IEEE Internet of Things Journal.

[20] Xie, Y., Li, F. and Wang, Y., 2023. FingerSlid: Towards Finger-sliding Continuous Authentication on Smart Devices via Vibration. IEEE Transactions on Mobile Computing.

[21] Li, Y., Liu, L., Deng, S., Qin, H., El-Yacoubi, M.A. and Zhou, G., 2023. Memory-augmented autoencoder based continuous authentication on smartphones with conditional transformer gans. IEEE Transactions on Mobile Computing.

[22] Li, Y., Sun, X., Yang, Z. and Huang, H., 2024. SNNAuth: Sensor-Based Continuous Authentication On Smartphones Using Spiking Neural Networks. IEEE Internet of Things Journal.

[23] Qing Yang, Ge Peng, David T. Nguyen, Xin Qi, Gang Zhou, Zdeňka Sitová, Paolo Gasti, and Kiran S. Balagani. A Multimodal Data Set for Evaluating Continuous Authentication Performance in Smartphones. In Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems (SenSys '14). ACM, New York, NY, USA, 358-359. DOI: 10.1145/2668332.2668366

2. https://developer.android.com/studio/profile

[24] Gattulli, V., Impedovo, D., Pirlo, G. et al. Touch events and human activities for continuous authentication via smartphone. Sci Rep 13, 10515 (2023). https://doi.org/10.1038/s41598-023-36780-3

[25] Schroff, F., Kalenichenko, D. and Philbin, J., 2015. Facenet: A unified embedding for face recognition and clustering. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 815-823).

[26] ARISOY, M.V., 2021. Signature verification using Siamese neural network one-shot LEARNING. International Journal of Engineering and Innovative Research, 3(3), pp.248-260.

[27] Anand, P., Singh, A.K., Srivastava, S. and Lall, B., 2019. Few shot speaker recognition using deep neural networks. arXiv preprint arXiv:1904.08775.

[28] Langvardt, K., 2019. Regulating habit-forming technology. Fordham L. Rev., 88, p.129.