International Journal of Computing and Digital Systems
2024, VOL. 17, NO. 1, 1–10

http://dx.doi.org/10.12785/ijcds/XXXXXX

LUCIDS: An Ultra-lightweight Swipe Characteristics Based Continuous Authentication Framework for Smartphones and Tablets

Rupanka Bhuyan*[1,2] and S. Thiyagarajan[2]

[1]*The ICFAI University, Nagaland, Chümoukedima, Nagaland, India*
[2]*Department of Computer Science, St. Joseph University, Chümoukedima, Nagaland, India*

Received 12 March 2024, Revised 15 June 2024, Accepted 10 July 2024

Abstract: Smartphones and tablets are convenient and popular portable devices with processing power and memory comparable to laptops and desktops. Their connectivity and networking capabilities add to their appeal. Operated through a touchscreen interface, these devices are highly user-friendly. Users store important data and apps for online shopping, banking, and security system monitoring on them. Typically, passwords, pattern locks, face-scans or PINs are used to protect these devices. However, after logging in if left unattended, these devices can be easily accessed by an intruder. Therefore, there is a need for a secondary security mechanism to maintain authentication even after the user logs in. And it is desirable to do this without requiring extra user attention. This paper discusses Continuous Authentication (CA) methods for both tablets and smartphones, which provide continuous and unobtrusive user authentication as a secondary security measure. An unobtrusive ultra-lightweight framework is introduced to implement this using a 117 users' dataset. The authentication performance of the framework was compared with state-of-the-art methods which use smartphones and also those using tablets. The results were competitive in both the scenarios. The framework uses only a single stroke swipe data for authentication eliminating any need for additional sensors data and providing detection at the earliest.

Keywords: smartphones, tablets, swipes, touch based continuous authentication, unobtrusive continuous authentication

1. Introduction

Smartphones and tablets are widely used for personal data storage and on-the-go information processing, with processing power similar to laptops. Their portability makes them convenient alternatives to laptops, serving various functions such as data repositories, authentication devices, payment tools, entertainment, security controls, and more [1]. These devices typically use touchscreens, which serve as both display and input interfaces, enhancing design and space efficiency. Due to their widespread use, robust security measures are essential to distinguish between legitimate and illegitimate users, preventing unauthorized access. Security can be addressed in two ways: first, through initial user authentication via methods like biometrics, passwords, or face scans; and second, through continuous or intermittent verification during device use, which helps prevent unauthorized access [1].

2. Touch Based Continuous Authentication (TCA)

Touch-based Continuous Authentication (TCA) refers to the ongoing verification of a user's identity on a tablet or smartphone after the initial login. This process relies on the user's touch interactions, such as tapping, typing, and swiping, which fall under behavioral biometrics. Each user's touch behavior is unique, allowing for the creation of a digital signature or touch behavioral profile [1]. TCA can function through various methods, including (a) drawn patterns, (b) fingerprints, (c) tap/type phrases, (d) touch or tap Captchas, and (e) Passwords/PINs/OTPs.

A. Importance of TCA

The importance of Touch-based Continuous Authentication (TCA) as a secondary defense for portable devices is evident in this scenario: After logging in via password, pattern, facial recognition, or fingerprint, a legitimate user leaves the device unattended and unlocked. This leaves the device vulnerable to unauthorized access. With TCA, such threats can be detected and blocked immediately.

TCA implementation requires no extra hardware, utilizing casual user inputs for data collection. Unlike biometric methods, compromised TCA profiles can be easily replaced. However, some TCA approaches may be obtrusive,

E-mail address: rupanka@yahoo.com, tyagarajin@yahoo.com

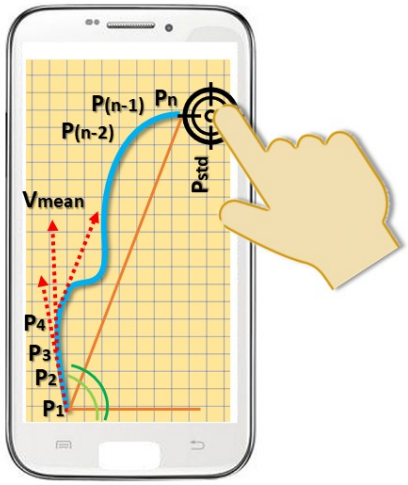

Figure 1. Swipe Characteristics on Touchscreen

requiring explicit user actions. This paper introduces an unobtrusive TCA method. Additionally, TCA should detect impostors quickly, preventing malicious activities.

B. *Swipes*

The most common activity that the users of these devices perform are the swipes on its touch screen and it greatly influences users' experience [2]. Technically, a swipe can be defined as an input modality wherein the user makes contact with the touchscreen of the device with a finite sequence of consecutive points, say $p_1, p_2, \ldots, p_n$. Each of these swipes has characteristics such as starting point, velocity, area, pressure, ending point, etc. [3][4].

3. Literature Review

Although, the number of research works conducted in the domain of touchscreen-based tablets with regard to continuous authentication is very few, a good number of research works based on swipe characteristics on touchscreen-based phones have been conducted. Some of the works uses sensor data also along with the touch characteristics. Due to its high relevance across different kinds of touchscreen devices, all these swipes' characteristics-based research works for both phones and tablets have been discussed below.

In the works of Naji et. al. [3], a Convolutional Neural Network based continuous authentication system is used which authenticates swipe gestures of 55 users of mobile devices resulting in accuracy of 92.24%, recall of 90.2%, precision of 84.71% and F1 score of 86.93%, respectively.

Mallet et. al. [5] applied machine learning algorithms, namely – Random Forest, Support Vector Machine, and K-Nearest Neighbor on the touch dynamics and phone movement datasets of 100 users and obtained the best authentication performances for the Random Forest as 81.74% in terms of accuracy, 63.06% in terms of precision, 97.35% in terms of recall, and 75.34% in terms of F1 score.

Machine Learning binary classifiers including Random Forest, K-Nearest Neighbor, and Support Vector Machine were employed in the work of Nascimento et. al. [6] to determine the authenticity of specific user actions based on touch dynamics in a dataset of 15 tablet users. Accuracy scores ranging from 78 - 90%, precision scores ranging from 88 - 95%, recall scores ranging between 88 - 94%, and F1 Scores in the range of 88 - 94% were obtained.

A Random Forest based method was introduced by Gattulli et. al. [7] for continuous user verification while using a smartphone and to identify illegitimate users during a reading activity. The works was carried out on a dataset of 20 users comprising of data from touch and hardware sensors. An accuracy of 96% was obtained while the F1 Score was 95%.

In a work by Aaby et. al. [8] the Extra Trees Classifier and Gradient Boosting algorithms were used on a 35-user dataset. While the accuracy of the method is not known, it could identify a user by using at least 3 strokes.

A related work by DeRidder et. al. [9] demonstrates the effectiveness of touch dynamics for user authentication using ML classifiers – Random Forest and K-Nearest Neighbor on a dataset of 25 users playing mobile games. The Random Forest algorithm performed better than K-Nearest Neighbor with accuracy of 93.45%, precision of 90.76%, recall of 86.74%, and F1 Score of 88.58%.

An experiment involving 40 participants used a Neural Network based technique resulting in accuracies of 90.04% and F1 Score of 91.41% in the works of Pelto et. al. [10].

A user authentication method on multiple smart devices using behavioral biometrics, combining Convolutional Neural Network and Long Short-Term Memory network was illustrated in the exploits of Wang et. al. [11] involving a dataset of 20 users. Sensor data from accelerometer and gyroscope were used for both smartphones and tablets resulting in accuracies of 99.8% and 99.2%, respectively. However, authentication could be done only when the user is walking.

The works of Li et. al. [12] introduces SearchAuth which is a continuous authentication system on smartphones using Neural Architecture Search and Auto Augmentation Search to enhance security. The technique utilizes accelerometer, gyroscope, and magnetometer sensor data from 88 users. Accuracy of 93.95% and F1 Score of 94.30% are reported for the work.

It is observed that the number of users involved in most

TABLE I. The Dataset

#	Parameter	Description
1	min_x	Minimum x coordinate within the full swipe.
2	min_y	Minimum y coordinate within the full swipe.
2	min_y	Minimum y coordinate within the full swipe.
3	max_x	Maximum x coordinate within the full swipe.
4	max_y	Maximum y coordinate within the full swipe.
5	$euclid_{dist}$	Euclidean Distance (ED) between the swipe's first and last points.
6	D_{list}	List of EDs between consecutive pairs of points composing a swipe.
7	tan_{angle}	The swipe's tangent angle.
8	tot_{time}	The time duration for the full swipe to complete.
9	v_{mean}	The swipe velocity's mean value.
10	v_{std}	The swipe velocity's standard deviation.
11	$v_{quarts}0$	The first quartile of velocity during the swipe.
12	$v_{quarts}1$	The second quartile of velocity during the swipe.
13	$v_{quarts}2$	The third quartile of velocity during the swipe.
14	a_{mean}	The swipe's mean acceleration.
15	a_{std}	The standard deviation of acceleration during the swipe.
16	$a_{quarts}0$	The first quartile of acceleration during the swipe.
17	$a_{quarts}1$	The second quartile of acceleration during the swipe.
18	$a_{quarts}2$	The third quartile of acceleration during the swipe.
19	p_{mean}	The swipe's mean pressure.
20	p_{std}	The swipe's pressure value's standard deviation.
21	$p_{quarts}0$	The first quartile of swipe's pressure.
22	$p_{quarts}1$	The second quartile of swipe's pressure.
23	$p_{quarts}2$	The third quartile of swipe's pressure.
24	$area_{mean}$	The mean value of the swipe's area.
25	$area_{std}$	The standard deviation of swipe's area.
26	$area_{quarts}0$	The first quartile of swipe's area.
27	$area_{quarts}1$	The second quartiles of swipe's area.
28	$area_{quarts}2$	The third quartiles of swipe's area.
29	$swipe_{type}$	The direction of the swipe (left, up, right, down).

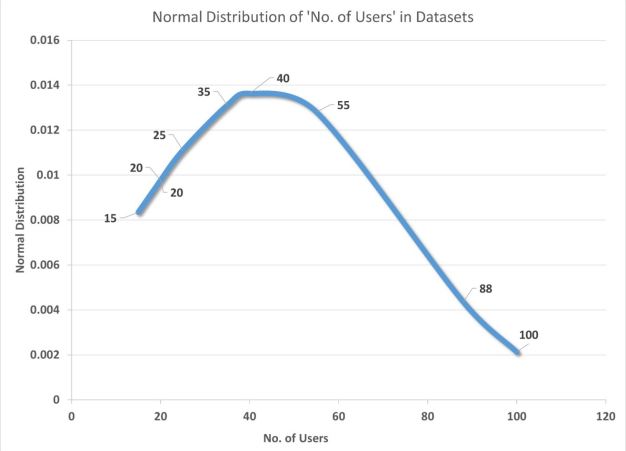

Figure 2. Distribution of No. of Users

of the works are few. The normal distribution peaks at a little over 40 as depicted in Figure 2.

4. Background Concepts

Before introducing the proposed methodology, the subsections below provide a theoretical background on the concepts discussed in the paper. This involves (i) introduction to the dataset, (ii) standardization of the data, (iii) machine learning methods, and (iv) performance measures used.

A. The Dataset

Although many datasets in this research area are not publicly accessible, some are available from their sources. Most of the datasets used in the contemporary works discussed in section 3, except one [11] uses only one type of device, i.e. either smartphone or tablet. This work uses a dataset, by Belman [13], which contains data from both smartphone and tablet users, respectively.

For example, Belman [13] recently introduced a dataset that includes data from both smartphones and tablets, detailed in Table I. This dataset comprises around 1,14,660 parameter values from 15,027 swipe vectors, encompassing 29 parameters collected from 117 users. The dataset is available at the url http://dx.doi.org/10.21227/rpaz-0h66

B. The Methods

Based on their performance and relevance to the problem at hand, seven methods have been identified. The next section discusses the fundamental working principles of these methods.

1) Support Vector Machine (SVM):

To perform binary classification wherein data samples are represented as points in space normally as

$$T = \{(x_i, y_i) : 1 \leq i \leq m\} \tag{1}$$

where x_i is an n-dimensional vector and $y_i \in +1, -1$ [14][15][16]], one of the methods is the Support Vector Machine (SVM). SVM creates a gap or distance between two categories. It classifies new or unclassified data points based on which side of the gap they fall on, with +1 for the positive side and −1 for the negative side. A hyperplane, which maintains the largest distance from the nearest points of each class, separates the two categories. These nearest points are known as support vectors [14][15][16]. Below, in equation (2), the classification function is represented in the following manner

$$f(x) = x \cdot w - B \tag{2}$$

with weight vector (w) and bias (B). During the training process, the values for w and B are computed. To correctly classify the training set [14][15][16], the function f must produce positive values for positive data samples and negative values for others. This means that the equations in (3) must be satisfied for all data points x_i in T. SVM identifies a linear separating hyperplane with the largest margin in this higher-dimensional space [14][15][16].

$$\begin{aligned} x_i \cdot w - B > 0, \quad if \quad y_i = +1 \\ x_i \cdot w - B < 0, \quad if \quad y_i = -1 \end{aligned} \tag{3}$$

Except in ideal conditions, an SVM by default may or may not yield the desired performance which is why certain parameter in the same may be optimized. The 'penalty' (usually denoted as C) and the 'type of kernel' being used are two such parameters which have been optimized for this work.

2) Decision Tree (DT):

Decision Trees (DTs) are supervised learning methods used for both classification and regression by learning decision rules from training data to predict the target variable's class or value. A DT consists of nodes containing data, with the topmost node being the root [14][15][16][17]. If the class label is unknown, the attribute values of a tuple T are compared to the DT, and the class prediction is found in a leaf node reached by following a path from the root. While constructing the DT by splitting nodes, two concepts are used: Entropy and Information Gain. Entropy measures the homogeneity of the dataset, indicating its degree of impurity and heterogeneity. For a dataset like T_{ORG} with positive and negative examples, equation (4) calculates the entropy relative to this Boolean categorization [18].

$$E(T_{ORG}) = -(p_{pos} Log_2 p_{pos} + p_{neg} Log_2 p_{neg}) \tag{4}$$

where p_{pos} and p_{neg} are the portions of positive and negative instances, respectively. Entropy measures dataset impurity, and the effectiveness of an attribute in classifying the training set can be quantified using Information Gain (IG). IG represents the reduction in Entropy achieved by splitting the dataset based on an attribute. Equation (5) defines the Information Gain $IG(T_{ORG}, A)$ for an attribute A in relation to the dataset T_{ORG} [14][15][16][17].

$$IG(T_{ORG}, A) = E(T_{ORG}) - \Sigma_{v \in Values(A)} \frac{|T^v_{ORG}|}{|T_{ORG}|} \tag{5}$$

where Values (A) represents the complete set of possible values for attribute A, T^v_{ORG} is the subset of T_{ORG} for which attribute A has value 'v'.

3) Artificial Neural Network (ANN):

Artificial Neural Networks (ANNs), as defined by Raschka [14], Grus [16], and Zheng [17], are mathematical constructs consisting of interconnected 'neurons'. The input layer receives parameter values x_i $(1 \leq i \leq n)$ from the dataset, which are then passed to one or more hidden layers. Each neuron in these layers has an activation function and receives weighted inputs from the previous layer. The neuron sums the weighted inputs, and if this sum exceeds a threshold τ, it outputs 1; otherwise, it outputs 0. The final layer, the output layer, receives inputs from the last hidden layer and provides the network's final output, which can be either singular or multiple. Furnished below in equation (8) is the mathematical model of a neuron given in [14][16][17]

$$y = \theta(\Sigma^n_{i=1} = w_i x_i - \tau) \tag{6}$$

The NN learns from the training samples in iterations and the performance of the network may be optimized using its multiple number of parameters.

4) Extremely Randomized Trees (ERT / ETC):

The Extra Trees Classifier (ETC) or Extremely Randomized Trees (ERT) ensemble technique generates predictions by averaging the outcomes of multiple randomly created trees [19]. Each tree is constructed by selecting the split that maximizes the splitting score from a set number of random splits at each node, with both the threshold τ and variable x_i chosen randomly. Trees are built iteratively until every leaf node and end node contains learning samples

below a predetermined threshold n_{min} or is pure in terms of outputs [19].

5) Adaboost (ADB):

Machine learning ensemble methods use the AdaBoost algorithm, or Adaptive Boosting (ADB), which redistributes weights so that incorrectly classified instances receive higher weights. This helps reduce bias and variation in supervised learning. During training, AdaBoost creates n Decision Trees (DTs), with each subsequent tree focusing on the records that were misclassified by previous trees. This process continues until the desired number of base learners is achieved [14][15][19].

6) Random Forest (RF):

The Random Forest (RF) algorithm may be described as a classifier made up of a number of tree structured classifiers $\{h(x, \theta_k), k = 1, \ldots\}$ where the θ_k are independent and identically distributed random vectors wherein each tree casts a single vote for the most prevalent class at input x [14][15][20]. The algorithm operates as follows: (i) n random records are selected at random from a data set containing k records; (ii) individual Decision Trees (DT) are built for each sample; (iii) each DT produces an output; and (iv) the final output is evaluated based on majority voting (for classification) and averaging (for regression), respectively.

7) Extreme Gradient Boost (XGB):

Extreme Gradient Boosting (XGB) [14][19] is a tree-based supervised machine learning method used for both classification and regression. It utilizes decision trees as base estimators, built using residuals. XGBoost's unique tree-building approach involves determining optimal node splits through the Similarity Score (SS) and Gain, as shown in equations (7) and (8).

$$SS = \frac{(\Sigma_{i=1}^{n} Residual_i)^2}{\Sigma_{i=1}^{n}[PP_i * (1 - PP_i)] + \lambda} \tag{7}$$

where, (i) Residual is the real value that was seen or anticipated, (ii) the likelihood of an occurrence determined in a prior stage is known as the Previous Probability (PP), (iii) Lambda is a parameter for Regularization.

The Gain is calculated using the following formula after obtaining the Similarity Score for each leaf as shown in equation (10) with Left Leaf (LL), is Right Leaf (RL) and Root (RT), respectively.

$$Gain = LL_{similarity} + RL_{similarity} - RT_{similarity} \tag{8}$$

C. Performance Metrics

While many studies focus on accuracy and equal error rate in relation to authentication performance, additional metrics are also important for a complete assessment. This includes evaluating how well the system flags genuine users as impostors or intruders as legitimate users. Classification concepts introduced in [14] are defined below.

True Positives (T^+): The accurately identified class samples are known as True Positives.

True Negatives (T^-) Correctly detected samples that do not belong in the class are referred to as True Negatives.

False Positives (F^+): False Positives are the quantity of samples that were incorrectly identified as belonging to the positive group.

False Negatives (F^-): Samples that were mistakenly assigned to the negative class whereas, in reality, they belong to the positive class.

Based on the above, this work lists the related metrics below in equations (9), (10), (11) and (12) which are used for measuring the overall performance of the CA technique.

$$Accuracy\,(A) = \frac{(T^+ + T^-)}{(T^+ + T^- + F^+ + F^-)} \tag{9}$$

$$Recall\,(R) = \frac{T^+}{(T^+ + F^-)} \tag{10}$$

$$Precision\,(P) = \frac{T^+}{(T^+ + F^+)} \tag{11}$$

$$F1\,Score = \frac{(2 * P * R)}{(P + R)} \tag{12}$$

5. Proposed Framework

The proposed framework involves a series of experiments described below in subsection 5.2. However, prior to that, the data are to be processed before being inputted to the various machine learning techniques.

A. Data Processing

To address scale differences in the original dataset, either $z - score$ or normalization techniques are applied [21]. $Z - score$ adjusts parameter values to a 0 to 1 range, while normalization scales them between -1 and $+1$. This process eliminates the bias from parameters with wider value ranges, ensuring uniformity. Such consistency is crucial for unbiased classification results in most machine learning methods. Equations (13) and (14) in [21] show how $z-score$ and normalized values are calculated. The $z-score$ value s_i for a parameter value x_i of a parameter x is given by

$$s_i = \frac{(x_i - M)}{S} \tag{13}$$

where, the symbol S represents the standard deviation while M is the mean value of x.

The normalized value n_i of a given parameter value x_i (belonging to a parameter x having minimum and maximum values x_{min} and x_{max}) is given by

$$n_i = \frac{(x_i - x_{min})}{(x_{Max} - x_{min})} \tag{14}$$

The data in the dataset need to be standardized before further processing.

B. Parameter B_k

In the original swipes portion of the BB-MAS dataset denoted in Table 1, there is a parameter named D_{list} which is explained below.

Let, (x_1, y_1), (x_2, y_2), ..., (x_p, y_p) be the consecutive coordinates composing a single 'swipe'. Then, the Euclidean distance [17] between any two consecutive points (x_d, y_d), (x_{d+1}, y_{d+1}) is given by

$$e = \sqrt{(x_d - x_{d+1})^2 + (y_d - y_{d+1})^2} \tag{15}$$

Based on the above concepts and equation (15), parameter D_{list} is defined below by the equation (16)

$$D_{list} = \{e_1, e_2, ..., e_{p-1}\} \tag{16}$$

where e_1 = Euclidean distance between (x_1, y_1) and (x_2, y_2), e_2 = Euclidean distance between (x_2, y_2) and (x_3, y_3), and so on. Based on the above concepts, a parameter named Bulk (B_k) has been proposed as equation (17)

$$B_k = 1 + |Dlist| \tag{17}$$

C. The Experiments

Initially, the original standardized swipe data of all the N number of users of the device (smartphones and tablets, separately) is considered as D_{ORG}. Subsequently a new parameter B_k is derived which is detailed in the next subsection below. A new dataset D_{AUG} is created from the concatenation of D_{ORG} and B_k. Consider U as the total set of N distinct number of users. For each user $u_j \in U$, an intruder dataset I_j is created from the $(N-1)$ other users. A set of 7 machine learning techniques discussed in the previous section 4, are applied on each of these u_j users using 10-fold cross validation [17][20]. The average of the results for the performance parameters (Accuracy - A, Precision - P, Recall - R and $F1$ score) are collected in set V. The experiment is repeated for D_{AUG} and the average of the performance parameters are collected in V. Figure 3 illustrates the data processing and proposed methodology.

6. The Algorithm

An algorithm is proposed to implement the LUCIDS framework for the experiments and is detailed below as Algorithm 1.

Algorithm 1: LUCIDS Implementation

Define T_{ORG} = original swipes dataset for tablet users
Define T_{AUG} = T_{ORG} augmented with parameter B_K
Define P_{ORG} = original swipes dataset for smartphone users
Define P_{AUG} = P_{ORG} augmented with parameter B_K
/* The steps 5 - 8, 12 and 17 will be applied for $D_X = T_{ORG}$, $D_X = T_{AUG}$, $D_X = P_{ORG}$ and $D_X = P_{AUG}$ in sequence */
Define $M = \{m_i : 1 \leq i \leq 7\}$ /* The set of seven methods */
Define V = $\{v_i$: v_i is a tuple (v_a, v_b, v_c, v_d) where v_a is the accuracy score of m_i, v_b is the precision score of m_i, v_c is the recall score of m_i, and v_d is the F1 score of m_i $\}$
Define $U = \{u_j : 1 \leq j \leq N\}$ where the distinct number of users in D_X is N
/* Set of Users */
for $j = 1$ **to** N **do**
    Define $U_j = \{x : x \in u_j$ and $U_j \subset D_X\}$
    /* Set of swipes for user u_j */
    Define $I_j = \{r : r \in D_X$ and $r \notin U_j\}$
    /* Intruder dataset I_j for user u_j contains n tuples where $n = (N-1) \times K$ and $K = \left\lfloor \frac{\text{no. of } u_j \text{ tuples in } D_X}{N-1} \right\rfloor$ */
    Define $D_j = U_j \cup I_j$
    /* Experimental dataset for user u_j */
for $i = 1$ **to** 7 **do**
    for $j = 1$ **to** N **do**
        Do m_i on D_j with 10-Fold Cross Validation
        Calculate pm_{avg} for method m_i
        /* Average of performance metrics */
        Set $v_i = pm_{\text{avg}}$
Output V

7. Results and Discussions

The algorithm from Section 6 was applied to four datasets: T_{ORG}, T_{AUG}, P_{ORG}, and P_{AUG} in four separate experiments labeled as Sets 1, 2, 3, and 4. T_{ORG} and P_{ORG} were referred to as D_{ORG}, while T_{AUG} and P_{AUG} were

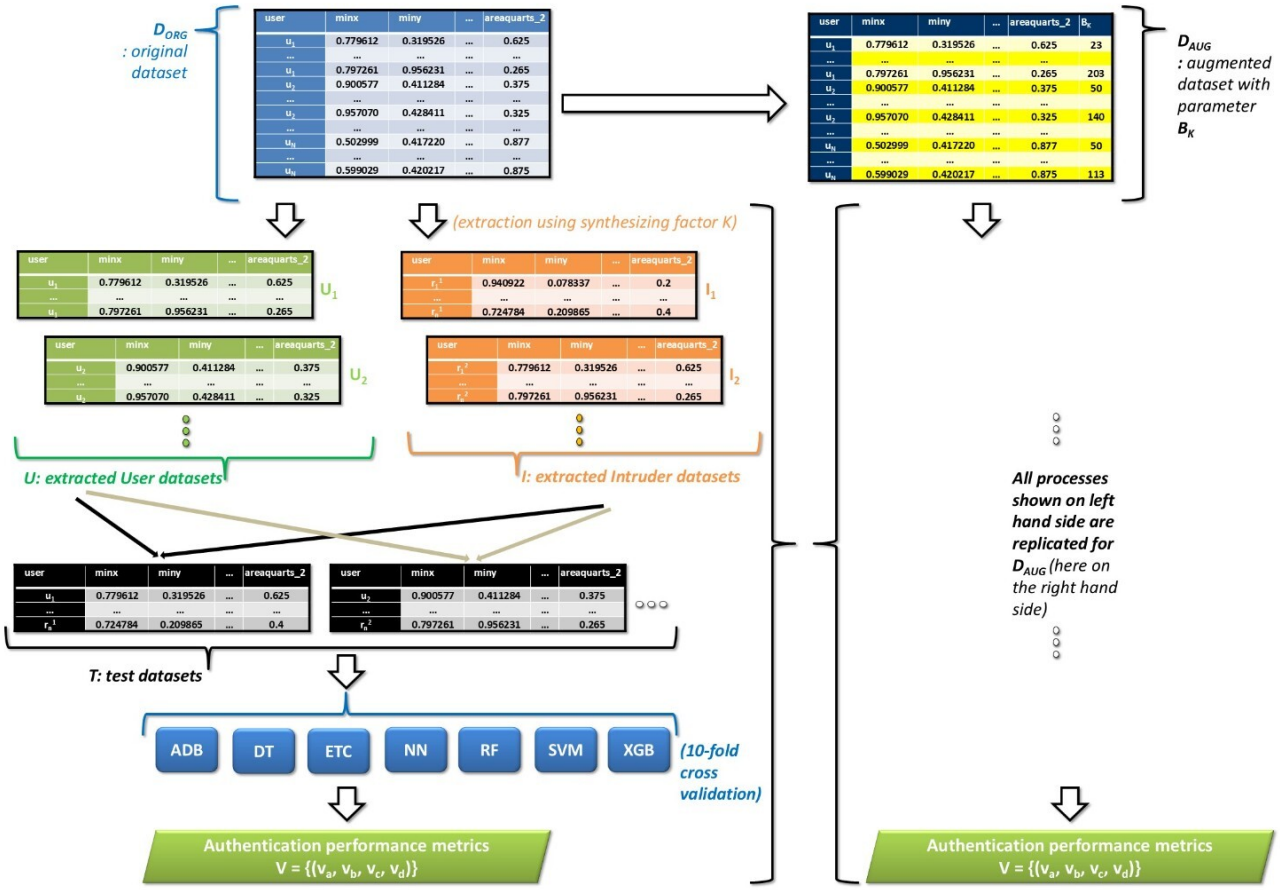

Figure 3. Implementation of the LUCIDS framework

termed D_{AUG}. The algorithm generated balanced datasets, D_js, using factor K, and applied 10-fold cross-validation [17][20] across 7 methods.

Experiment Set 1 (T_{ORG}) achieved the highest accuracy (92.07%) and F1 score (93.04%) with Extreme Gradient Boost (XGB). Set 2 (T_{AUG}) with B_K augmentation also saw XGB excel, with 92.17% accuracy and 93.11% F1 score. In Set 3 (P_{ORG}), XGB led in accuracy (93.77%), and AdaBoost (ADB) in F1 (93.61%). Set 4 (P_{AUG}) saw the Support Vector Machine (SVM) dominate, with 94.32% accuracy and 94.39% F1 score. Table II presents these results.

Overall, augmenting datasets with B_K improved performance, with XGB excelling for tablets and SVM for smartphones. Figures 4, 5, 6 and 7 shows the comparative performance results.

Figure 4. Performance on dataset T_{ORG}

8. Improvements over the State-of-the-Art Methods

The proposed LUCIDS framework resulted in the following improvements over the State-of-the-Art methods:

(1) both smartphones and tablet devices are included in this work which most of the contemporary works did not include.

(2) authentication is done without requiring any addi-

TABLE II. Experiment Sets

#	Dataset	Method	Accuracy	F1 Score
Set 1 : Tablets (original dataset)				
1	T_{ORG}	ADB	0.9160	0.9250
2	T_{ORG}	DT	0.8399	0.8519
3	T_{ORG}	ETC	0.9149	0.9251
4	T_{ORG}	NN	0.8821	0.8953
5	T_{ORG}	RF	0.8939	0.9023
6	T_{ORG}	SVM	0.8951	0.9095
7	T_{ORG}	XGB	0.9207	0.9304
Set 2 : Tablets (dataset with parameter B_K)				
8	T_{AUG}	ADB	0.9161	0.9247
9	T_{AUG}	DT	0.8411	0.8513
10	T_{AUG}	ETC	0.9161	0.9248
11	T_{AUG}	NN	0.8864	0.9007
12	T_{AUG}	RF	0.8905	0.8967
13	T_{AUG}	SVM	0.8960	0.9113
14	T_{AUG}	XGB	0.9217	0.9311
Set 3 : Phones (original dataset)				
15	P_{ORG}	ADB	0.9376	0.9361
16	P_{ORG}	DT	0.8832	0.8795
17	P_{ORG}	ETC	0.9337	0.9309
18	P_{ORG}	NN	0.9333	0.9326
19	P_{ORG}	RF	0.9196	0.9134
20	P_{ORG}	SVM	0.9322	0.9321
21	P_{ORG}	XGB	0.9377	0.9359
Set 4 : Phones (dataset with parameter B_K)				
22	P_{AUG}	ADB	0.9369	0.9352
23	P_{AUG}	DT	0.8838	0.8808
24	P_{AUG}	ETC	0.9348	0.9320
25	P_{AUG}	NN	0.9342	0.9334
26	P_{AUG}	RF	0.9172	0.9106
27	P_{AUG}	SVM	0.9432	0.9439
28	P_{AUG}	XGB	0.9372	0.9350

Figure 5. Performance on dataset T_{AUG}

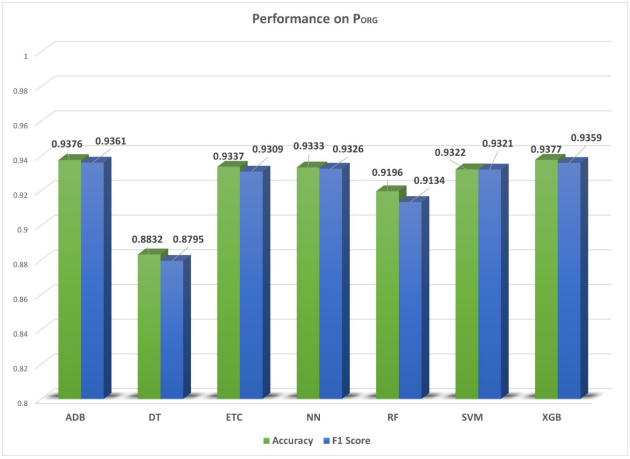

Figure 6. Performance on dataset P_{ORG}

Figure 7. Performance on dataset P_{AUG}

tional sensors' data because the method does not require the user to perform any additional activities like walking, reading, etc.

(3) most of the contemporary methods have used datasets with a few number of users. The peak in the Normal Distribution curve for the same spikes near above 40 (Figure II). This work uses over 100 numbers of users.

(4) authentication is done using only a single swipe from the user. The ensures intruder detection at the earliest before the intruder is able to perform any further activities.

Separately, a comparison of LUCIDS with the State-of-the-Art methods are described in Table III.

The basic comparatives between State-of-the-Art methods and LUCIDS in terms of No. of Users, Accuracy and F1 score is illustrated in Figure 5.

#	Work	Method	Users	Data Description	Performance	Remarks
1	Naji [3]	CNN based continuous authentication system	55	Swipe gestures; Subsets of BioIdent and HMOG datasets	92.24% (Acc) 86.93% (F1)	- works in Smartphones
2	Mallet [5]	RF, SVM, and KNN based methods	100	Touch dynamics / phone movement data from BioIdent and HMOG datasets	81.74% (Acc) 75.34% (F1)	- works in Smartphones - Sensor data required
3	Nasci-mento [6]	RF, KNN, and SVM based classifiers	15	Customized dataset on touch actions of user	78-90% (Acc) 88-94% (F1)	- works in Tablets
4	Gattulli [7]	RF based method for continuous user verification	20	Subset of HMOG dataset	96.00% (Acc) 95.00% (F1)	- works in Smartphones - works while reading
5	Aaby [8]	ETC and Gradient Boosting based methods	35	Subset of Touchalytics dataset	(Not mentioned)	- works in Smartphones - needs Min. 3 strokes
6	DeRidder [9]	RF and KNN based classifiers	25	Touch dynamics data from users playing mobile games	93.45% (Acc) 88.58% (F1)	- works in Smartphones
7	Pelto [10]	NN based technique	40	Data from participants playing mobile games	90.04% (Acc) 91.41% (F1)	- works in Smartphones
8	Wang [11]	Combination of CNN and LSTM for authentication when user is walking	20	Sensor data from accelerometer and gyroscope	Smartphones: 99.80% (Acc) Tablets: 99.20% (Acc)	- works in Smartphones - works in Tablets - sensor data required
9	Li [12]	SearchAuth: based on NN architecture	88	Utilizes accelerometer, gyro, and magnetometer sensor data	93.95% (Acc) 94.30% (F1)	- works in Smartphones - sensor data required
10	LUCIDS (This work)	Ultra-lightweight framework for swipes based continuous authentication	117	15,027 swipe vectors composed of 29 parameters from 117 users	Smartphones: 94.32% (Acc) 94.39% (F1) Tablets: 92.17% (Acc) 93.11% (F1)	- works in Smartphones - works in Tablets - needs only swipe action - Max. 1 swipe required - sensor data not required

TABLE III. Summary of State-of-the-Art vs. LUCIDS.

9. Conclusions and Future Work

This paper introduces LUCIDS, an ultra-lightweight framework for unobtrusively detecting legitimate or illegitimate users on smartphones and tablets during device use after login. It leverages the user's swipe characteristics on the device's touchscreen without requiring any additional or explicit gestures or activities (such as, walking or reading or playing games). This works as a secondary security

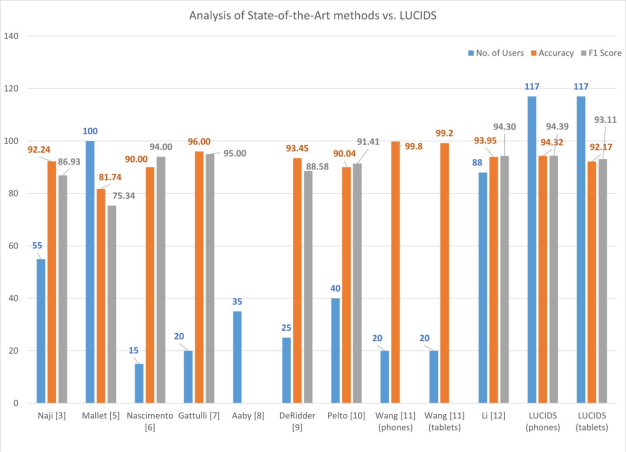

Figure 8. State-of-the-Art vs. LUCIDS

mechanism performing continuous authentication in the background based only on the user's swipe action.

Various methods were tested on both original and augmented swipe data incorporating a new parameter B_k. For tablets, Extreme Gradient Boost (XGB) outperformed other methods, while for smartphones, the Support Vector Machine (SVM) technique was more effective. The authentication results were comparable (94.32% accuracy for smartphones, 94.39% F1 score for smartphones, 92.17% accuracy for tablets, and 93.11% F1 score for tablets) in comparison to other State-of-the-Art methods. Some of the highlights of LUCIDS with respect to other State-of-the-Art works are, firstly, it uses the highest number of users (117); secondly, authentication is done within one single swipe action of the user; thirdly, both types of devices, namely smartphones and tablets are used; and lastly, it does not require the user to perform any other additional or explicit actions (eliminating the need for extra sensors data).

Future research in this area will explore the practicality of user profile portability across multiple touchscreen based devices.

10. Declarations

A. *Conflict of Interest*

The authors declare no conflict of interest.

B. *Acknowledgements*

The first author dedicates the variable B_k in the name of his father Late Mr. Kabindra Nath Bhuyan.

C. *Data Availability Statement*

All data that support the findings of this study are included in the article.

D. *Ethical Approval*

This research work does not require any ethical approval.

E. *Funding*

This research did not receive any specific grant from agencies in the public, commercial, or not-for-profit sectors.

References

[1] R. Bhuyan, S. P. K. Kenny, S. Borah, D. Mishra, and K. Das, "Recent advancements in continuous authentication techniques for mobile-touchscreen-based devices," in *Intelligent and Cloud Computing*, ser. Smart Innovation, Systems and Technologies, D. Mishra, R. Buyya, P. Mohapatra, and S. Patnaik, Eds. Singapore: Springer, 2021, vol. 194, pp. 263–273.

[2] Y. Zhu, "The role of touch, touchscreens, and haptic technology in interactive marketing: Evolution from physical touch to digital touch," in *The Palgrave Handbook of Interactive Marketing*, C. L. Wang, Ed. Cham: Palgrave Macmillan, 2023.

[3] Z. Naji and D. Bouzidi, "Deep learning approach for a dynamic swipe gestures based continuous authentication," in *Proceedings of The 3rd International Conference on Artificial Intelligence and Computer Vision (AICV)*, vol. 164. Springer, 2023, pp. 48–57.

[4] E. Ellavarason, R. Guest, and F. Deravi, "Evaluation of stability of swipe gesture authentication across usage scenarios of mobile device," *EURASIP Journal on Information Security*, vol. 2020, no. 4, 2020.

[5] J. Mallet, "Hold on and swipe: A touch-movement based continuous authentication schema based on machine learning," in *Proceedings of Asia Conference on Algorithms, Computing and Machine Learning (CACML)*, 2022.

[6] P. G. do Nascimento, P. Witiak, T. MacCallum, Z. Winterfeldt, and R. Dave, "Your device may know you better than you know yourself – continuous authentication on novel dataset using machine learning," *arXiv preprint arXiv:2403.03832*, 2024.

[7] V. Gattulli, D. Impedovo, G. Pirlo, and F. Volpe, "Touch events and human activities for continuous authentication via smartphone," *Dental Science Reports*, pp. 1–7, 2023.

[8] P. Aaby, M. V. Giuffrida, W. J. Buchanan, and Z. Tan, "An omnidirectional approach to touch-based continuous authentication," *Computers & Security*, vol. 128, p. 103146, 2023.

[9] Z. DeRidder, N. Siddiqui, T. Reither, R. Dave, B. Pelto, M. Vanamala, and N. Seliya, "Continuous user authentication using machine learning and multi-finger mobile touch dynamics with a novel dataset," in *9th International Conference on Soft Computing & Machine Intelligence (ISCMI)*, 2022, pp. 42–46.

[10] B. Pelto, M. Vanamala, and R. Dave, "Your identity is your behavior - continuous user authentication based on machine learning and touch dynamics," *arXiv.org*, 2023.

[11] Y. Wang, X. Zhang, and H. Hu, "Continuous user authentication on multiple smart devices," *Information*, 2023.

[12] Y. Li, J. Luo, S. Deng, and G. Zhou, "Searchauth: Neural architecture search-based continuous authentication using auto augmentation search," *ACM Transactions on Sensor Networks*, vol. 19, no. 4, pp. 1–23, November 2023.

[13] A. K. Belman, L. Wang, S. S. Iyengar, P. Sniatala, R. Wright, R. Dora, J. Baldwin, Z. Jin, and V. V. Phoha, "Insights from bb-mas

- a large dataset for typing, gait and swipes of the same person on desktop, tablet and phone," *arXiv:1912.02736*, 2019.

[14] S. Raschka, Y. Liu, and V. Mirjalili, *Machine Learning with PyTorch and Scikit-Learn*. Birmingham, UK: Packt Publishing, 2022.

[15] H. Tatsat, S. Puri, and B. Lookabaugh, *Machine Learning and Data Science Blueprints for Finance*. O'Reilly Media, 2020.

[16] J. Grus, *Data Science from Scratch: First Principles with Python*, 2nd ed. Sebastopol, CA, USA: O'Reilly Media, 2020.

[17] A. Zheng and A. Casari, *Feature Engineering for Machine Learning: Principles and Techniques for Data Scientists*, 1st ed. O'Reilly Media, Inc., 2018.

[18] O. Cherednichenko, D. Chernyshov, D. Sytnikov, and P. Sytnikova, "Generalizing machine learning evaluation through the integration of shannon entropy and rough set theory," in *Proceedings of the 8th International Conference on Computational Linguistics and Intelligent Systems, Machine Learning Workshop*, vol. 19.

[19] P. Bahad and P. Saxena, "Study of adaboost and gradient boosting algorithms for predictive analytics," in *Algorithms for Intelligent Systems*, G. S. Tomar, N. S. Chaudhari, J. L. V. Barbosa, and M. K. Aghwariya, Eds. Singapore: Springer, 2019.

[20] A. C. Müller and S. Guido, *Introduction to Machine Learning with Python: A Guide for Data Scientists*, 2nd ed. Sebastopol, CA: O'Reilly Media, 2021.

[21] R. Bhuyan, "Contemporary linear stochastic models for forecasting iot time series data," in *Lecture Notes in Networks and Systems*. Springer, 2020, pp. 99–106.