International Research Journal of
SCIENCE, TECHNOLOGY, EDUCATION, AND MANAGEMENT
P-ISSN: 2799-063X | E-ISSN: 2799-0648
ISO 9001:2015 Certified
Volume 3, No. 4 | December 2023

Improving network security using keyboard dynamics: A comparative study

Ugwunna, C.O.[1], Chukwuogo, O.E.[2], Alabi, O.A.[1], Kareem, M.K.[1], Belonwu, T.S.[2], Oloyede, S.O.[1]

[1] *Department of Computer Science, Federal University of Agriculture, Abeokuta, P.M.B. 2240, Abeokuta, Ogun State, Nigeria.*

[2]*Department of Computer Science, Nnamdi Azikiwe University Awka Anambra, P.M.B 5025, Awka, Anambra State, Nigeria.*

Corresponding email: ugwunnaco@funaab.edu.ng

ABSTRACT

Duplication or imitation of individual keystroke rhymes is very difficult which can make it very efficient to be used for identity authentication. Over time, it is possible that the keystroke style of an individual to be learned by following keystroke information obtained when the person types text. The user's identity can always be verified by studying the user's keyboard input styles anytime the user uses the keyboard. The technique suggested in this study uses the keystrokes that users make while typing to verify their identities. To provide an accurate verification of whether a user is authentic or fraudulent, a model that integrates machine learning and dynamic keystroke models—Decision Tree, Random Forest, Support Vector Machine, and K-nearest Neighbors—is compared and utilized. The keystroke dynamics dataset was gathered from Kaggle and consists of 51 subjects' keyboard dynamics data, which was collected over the course of eight sessions and six months. There are 20400 samples in all in the data. This study assessed the effectiveness of machine learning algorithms with a focus on the keystroke dynamic authentication system. Python is used for the development work, while Jupyter notebook is used as the IDE. The performance of the models for different variables is assessed using the following metrics: accuracy, error equal rate, parameter performance, threshold, training time, and testing time. According to the results, the accuracy of the Random Forest, Support Vector Machine, KNN, and Decision Tree algorithms are, respectively, 98, 97.55, 97.28, and 94.26%. Based on the comparing results, Random Forest outperforms the other models, suggesting that Random Forest can be used as the system model for Keystroke Dynamic authentication.

ARTICLEINFO

Received : Sept. 22, 2023
Revised : Nov. 28, 2023
Accepted : Dec. 30, 2023

KEYWORDS

Authentication, Keyboard Dynamics, Keystrokes, Machine Learning.

Suggested Citation (APA Style 7th Edition):

Ugwunna, C.O., Chukwuogo, O.E., Alabi, O.A., Kareem, M.K., Belonwu, T.S., & Oloyede S.O. (2023). Students' acceptance of learning management system: Analysis of responses based on generational differences. *International Research Journal of Science, Technology, Education, and Management,* 3(4), 104-121. https://doi.org/10.5281/zenodo.10516267

INTRODUCTION

Today's world is driven by information and the internet makes its communication seamless (Uschold & Gruninger, 2004). Unfortunately, by its nature, the Internet cannot guarantee the safety of the information it transmits (Goldberg & Wagner, 1996). In majority of cases, the primary means of user authentication in most computer systems is by user login ID and password which are typed in using the keyboard. Protecting this data keyed in using the keyboard is becoming increasingly difficult hence the need for cryptographic methods of protection which is the act of coding or concealing information so that only the intended receiver can read it. (Revett et al., 2007). The primary objective of cryptography is geared at providing information security services of data confidentiality, integrity and authenticity; the message is encrypted before they are stored and/or transmitted (Auyporn & Vong, 2015). According to Andrean et al. (2020), authenticating individuals is an essential part of web-based security since it prevents malevolent users from infiltrating the system. Despite the effort to protect the information either on storage and/or on transit, hackers and other security threats still successfully crack weak security measures such as passwords using either brute force or simply obtaining information through shoulder navigation. To counter this increase risk, there exist new systems that will differentiate one user from another that is differentiating between a fraudster and a genuine user, even if the correct credentials are typed in. One of these systems is Keyboard dynamics (Kim et al., 2018). Keystroke dynamics are a proposed way for enhanced user identification and authentication, as Sharma and Stamp (2023) describe it. Keystroke dynamics-based authentication (KDA) is a biometric user authentication technique based on a person's distinctive typing style. It uses their keystroke dynamics to identify them (Kim & Kang, 2020). Keyboard dynamics is easy to implement, inexpensive; it additionally offers the user some advantages in terms of model savings and typing speed and are discreet and clear. A standard/enhanced keyboard, mobile phone preferably a smart phone is required and the ability to control the delay between keystrokes while typing, which is compatible with all modern operating systems (Dorca-Josa, 2017).

When necessary, keystroke dynamics, a behavioral biometric, can be utilized to distinguish between genuine users and imposters (Eude & Chang, 2017). Keyboard dynamics is a behavioral biometric that is concerned with unique patterns and rhythms and timing created when an individual types on a keyboard (Krishnamoorthy et al., 2018). Keystroke dynamics, which take into account the typing habits of legitimate users, have been utilized to strengthen password-based user identification systems. Login-based authentication systems' primary flaw is their inability to identify users once they have already logged in (Kim et al., 2018). On a whole, it considers overall speed, peculiar mistakes why typing, duration of time that keyboard keys are depressed, and variations in speed when moving between specific keys. Keyboard dynamics refers to precise timing data that describes when each key is pressed (key down (KD)) and released (key up (KU)) while the user is typing on a computer keyboard. Approximate measures for keyboard dynamics are hold time (duration of keystrokes) and flight time (duration between keystrokes) and other variables include typing speed, error rate, numeric keypad, and modifier keys. The dynamic keystroke that this research focuses on makes use of the user's natural keyboard-typing pattern. Since each person's pattern is distinctive, it can serve as a reliable tool for categorization, continuous monitoring, identification, and verification. In contrast to other biometric technologies like fingerprint, voice, and hand geometry recognition (Kim et al., 2018). Keystroke dynamics has some validity since, according to Piugie et al. (2022), it's a simple method of enhancing password authentication security without requiring any intrusive user handling. Modern machine learning technology allows computers to learn even when they are not explicitly designed to accomplish so. Using test data and historical data, the computer predicts the outcomes of newly introduced data that it has never seen before, creating an algorithm that may be applied to future generalization (Bell, 2022). Machine learning could be any of these three: Supervised learning (where a labeled training data is used to develop the learning mapping function that turns the input variable (X) unto the corresponding output variable (Y) thereby solving for f in the mapping function $Y = f(X)$. The result is that it could accurately generalize an output when given a new input; (e.g. K-Nearest Neighbour, SVM, Decision Tree, and Random Forest). Unsupervised machine learning on the other hand, uses only unlabelled training data to predict the output (e.g. K-Means algorithm). Finally, machine learning could be by Reinforcement Learning where an agent decides the best next action to take based on its present state by learning behaviors that will maximize a reward; this learning is done by trial and error

(Arulkumaran, 2017). Since machine learning techniques promise improved security and accuracy, they are commonly used in authentication systems (Ryu et al., 2021).

A multilayer perception (MLP) was developed by Ahmed and Traore (2014) to automatically learn specific typing behavior patterns and combinations of typing functionalities. According to their observations, the EER was 2.46%. The learning algorithm facilitates the creation of customized and trained systems that produce accurate outcomes. The combination of dynamic key validation and machine learning enhances the ability to identify keyboard rhythms and predicts the legitimacy of an unidentified user. Based on features extracted from fixed-text keystrokes, Chang et al. (2022) explored a broad range of machine learning and deep learning techniques; the resulting model was improved and assessed against the findings of previous studies. It was discovered that multi-layer perceptron (MLP) and extreme gradient boosting (XGBoost) based models performed well. compared to analogous earlier research. In 2018, Dorca-Josa (2017) presented her thesis, which looked at how machine learning techniques were applied to enable button-touch authentication on mobile devices utilizing keyboard dynamics. The Android app iProfile gathers data from users who consented to enter the ".tie5Roanl" password five or six times every day. The gathered data was used to train three machine learning techniques: Random Forest, SVM-Linear, and SVM-Radial Basis Function (RBF). All three of the selected algorithms had higher than 97% classification accuracy, with random forest having the highest proportion. These results imply that machine learning could be used by dynamic key authentication to predict user authenticity (Dorca-Josa, 2017). A CNN and an RNN are combined in a model developed by Banerjee and Woodard (2012) to learn a sequence of unique keyboard vectors and generate unique keystroke features that provide identity authentication. Using two public datasets, the model's optimal FRR, FAR, and equal error rate (EER) are found to be (2.07%, 6.61%), (3.26%, 5.31%), and (2.67%, 5.97%), respectively. In the study of de-Marcos et al. (2021), the HMOG dataset—which documents 100 users' interactions over 24 sessions—was used to train seven distinct machine learning classifiers (eight reading sessions, eight writing sessions, and eight map navigation sessions). These techniques included probabilistic techniques (naive Bayes), instance-based techniques (k-NN), hyperplane optimization (SVM), decision trees (CART), and ensemble techniques (RFC, ETC, and GBC). The study suggested an agent model for creating and integrating CA in mobile devices. With an average accuracy of almost 0.70 for each and every crucial event, the ensemble algorithms from RFC, ETC, and GBC outperformed the others. GBC outperformed all other classifiers, and the differences were statistically significant. Using naive Bayes and k-NN, an accuracy of roughly 0.65 was attained. SVM performed the worst when compared to the other methods. Three distinct approaches for user identification while typing on a keyboard were proposed by Mondal and Bours (2016). There are several different machine learning algorithms, as well as a proposed and undisputed method of merging user pairs. These algorithms can be used alone or in combination with other methods. It was observed that, for users using both hands, the bottom-up structure theme achieved the most effective accuracy of 89.7%.

Revett et al. (2007) used a modified PNN with a small dataset of sample login IDs and passwords in their experiment. The researchers found that the modified classifier performed substantially better than the traditional PNN technique (4% versus 8%, approximately). The results of their investigation demonstrate that the attribute selection procedure and, to a lesser extent, the used authentication technique have a considerable impact on the Equal Error Rate (EER). The study's conclusions also demonstrate that a Probabilistic Neural Network (PNN) can outperform a regular MLFN back-propagation trained neural network in terms of training efficiency and classification accuracy. According to the study's findings, the PNN outperforms the MLFN in item classification and takes less training time. One of the biggest free text typing datasets was gathered by Alsuhibany et al. (2020), and it contained information on every software program, keyboard, and mouse operation. With the data set, they applied Gunetti & Picardi's algorithmic approach, yielding the most efficient EER of 10.36%. Huang et al. (2017) proposed an Algorithm Grain Density (KDE) rule that will be used to compute the distance between training and test samples using the probability density between training and test data sets in order to validate an individual's validity. The method was tested on multiple newly found datasets and achieved an EER of 1.95%. In the study by Pavaday and Soyjaudah (2007), a toolbox in Visual Basic 6.0 was constructed to facilitate data gathering as the user wrote on the keyboard. By tracking the keyboard inputs 1000 times per second, the method records how a user

types at a terminal. The equal error (EE) point, which may be used to compare the effectiveness of various matchers, is located at a distance of 2.95 and a pass rate of 65%.

Bhadri Naarayanan (2020) looks on the feasibility of exploiting user typing patterns on touchscreen keypads for the authentication process using free-text keystroke dynamics. Hold time, flight time, and digraph are the three timing features they used to build their timing vectors. Using Manhattan and Euclidian distances, the study calculated how similar the user's profile was to their login information. An Android app developed for the study may demonstrate how free texting impacts the accuracy of user authentication when utilizing a touch screen device. According to Maiorana et al. (2020), academics' interest in keystroke dynamics for biometric recognition has expanded since the introduction of mobile cellphones. In their paper, Antal and Nemes (2016) proposed a novel type of password termed logical strong, which refers to a password with a strong strength but is also easy to remember. Keystroke dynamics were employed in their work as a touchscreen-based device authentication method. The study looked at time-based, touch-based, and accelerometer-based data as the three main evaluation components. According to their findings, keystroke dynamics-based authentication methods are most suited for using strong passwords. The least equal error rate (EER) is obtained by the logical strong password, which is followed by the strong password, and the least effective is achieved by the simple password. The increasing significance of mobile devices in our daily lives served as another driving force for the research conducted by Draffin et al. (2014). a state-of-the-art passive authentication method that mimicked the way mobile users interact with cellphones' soft keyboards. According to the study, by using micro-behavior features without any contextual information, it is able to passively determine, within 5 keypresses, 67.7% of the time, that a mobile device is being used by an illegal user. The false acceptance rate (FAR) was 32.3%, and the false rejection rate (FRR) was only 4.6%, despite the fact that the detection rate after 15 keystrokes is 86% with a FAR of 14% and a FRR of only 2.2%.

Giot evaluated and presented a great deal of the keyboard dynamics research that has been done. Giot's study highlighted the problem of cross-devices, which refers to the use of the same device for data input. Real-time users can be using a range of gadgets with different keyboard layouts and screen locations. He claims that in order for the model to recognize users of different computing devices, it must be trained (Giot, 2009). Gurdal and Sogukpinar (2018) studied frequency key data by taking use of the movement of moving ridges, which enhanced the performance of a key-based authentication system The experimental findings of PIN authentication were found to be up to 21.8% higher than other strategies in the study conducted by Kim et al. (2020), and they also proposed a novel feature-based filter selection method in the context of dynamic authentication. In the study of Juola et al., a considerable corpus of keyboard behavior was also constructed using a simulated office setting (2013). Thanks to stylometric analysis, user distinction was feasible with a high degree of accuracy. An overview of two computerized behavioral biometric approaches was offered in the work of Bhatnagar et al. (2013). The analysis revealed that the variety of mouse types, mouse pads, and software configurations that could affect the verification's effectiveness is a major disadvantage of mouse-based verification over keyboard-based verification.

OBJECTIVES OF THE STUDY

This study aims to determine whether sample classification methods based on artificial intelligence can be utilized to identify or even validate the identities of computer system users.
The specific objectives of the study are as follows:
1. To identify illegal authentication.
2 Examine some of the literature on keyboard dynamics to comprehend the shortcomings of the current dynamic systems.
3. Compare and contrast various machine learning algorithms for enhancing network security
.

MATERIALS AND METHODS

This section proposes an authentication system that uses keyboard dynamics and machine learning for user's authentication. The proposed system model is divided into four main phases: data collection, training, testing, and evaluation. We collected necessary data, preprocessed the data so that they would be usable with our chosen machine learning algorithms and subsequently trained and tested the model so that we could use the model for future classification of users into valid and invalid users.

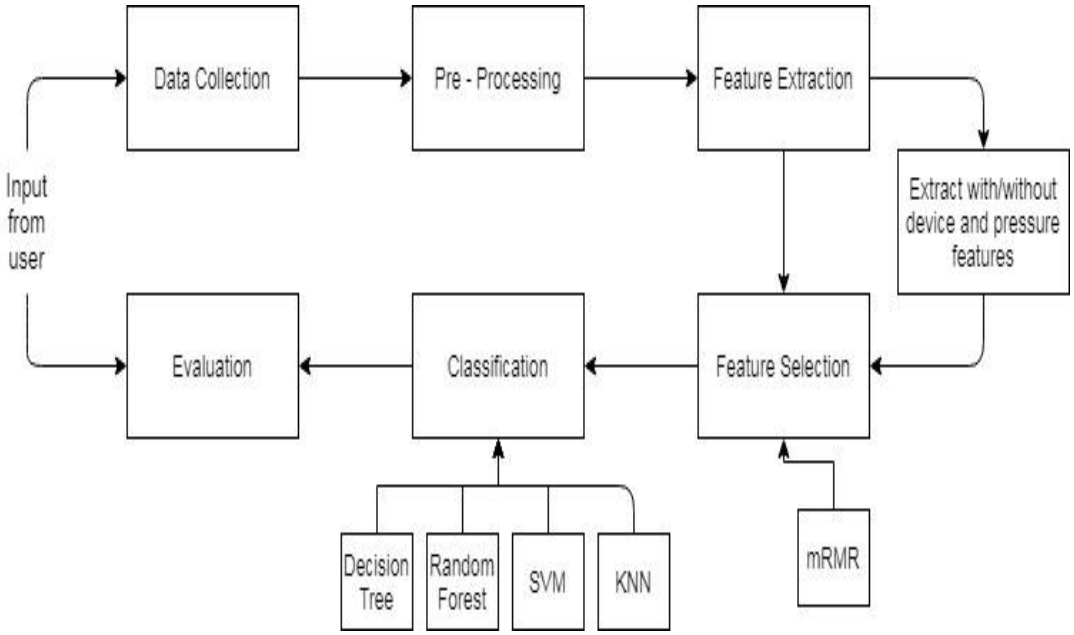

Figure 1: User Biometric Authentication Using Keyboard dynamics

The model can be created by following the stages as shown in figure 1.
- Data Gathering
- Prior to processing
- Features selection
- Features extraction
- Classification
- Evaluation

Gathering pertinent data and preparing it into a format that can be used for modeling are the first steps in solving machine learning difficulties. Pre-processing is necessary because of the data's standardization and scaling down. The raw dataset will be used to extract features. After the dataset has the class, the classifier will be run using the important features that were chosen using a feature selection algorithm. The present study employed Maximum Relevance – Minimum Redundancy (mRMR) as the feature selection algorithm, with Decision Tree, Random Forest, SVM, and K-Nearest Neighbors as the classifiers. By creating a model, a user can be identified and classed during the classification stage. Finally, an assessment was conducted to determine the efficacy (or lack thereof) of the algorithm.

Data Gathering

The Android software iProfile is used to gather keystroke events from Android devices and creates the dataset. To take part in the registration procedure, the user needs to install the program and grant permission. The user typed the same password six times per day for five days in a row into the iProfile program during the five-day data gathering phase. This process continued for six weeks. The iProfile program employs the password

"tie5Roanl," which combines capital and lowercase characters with numbers. The shift key can be used by the user to potentially travel between various keyboard layouts. There are 20400 samples in all in the data. The information is arranged into a table with 34 columns. The time data for one user repeating the password is represented by each row of data. Each subject's unique identifier (such as s002 or s057) is found in the first column. The identifiers in the dataset do not extend from s001 to s051, despite the fact that there are 51 people in total. Each subject received a unique ID for a variety of keystroke experiments, and not all subjects took part in every trial. For example, s001 does not exist in the data set since Subject 1 did not complete the password entering job. The password-typing session is indicated by the second column, sessionIndex, which has values between 1 and 8. Rep, the third column, *rep*, is the repetition of the password within the session (ranging from 1 to 50).

The timing details for the password are shown in the final 31 columns. The type of timing information is encoded in the column name. The column names in the form H.key indicate the hold time (i.e., the interval between pressing and releasing the key) for the designated key. The column names in the form DD.key1.key2 indicate the keydown-keydown time (i.e., the interval between pressing keys 1 and 2) for the designated digraph. The column names in the form UD.key1.key2 indicate the keyup-keydown time (i.e., the interval between pressing key 1 and pressing key 2) for the designated digraph. Keep in mind that UD times might be negative and that DD times are the sum of UD and H times.

Table 1: Keystroke Dataset Sample

Subject	sessionIndex	Rep	H.period	DD.period.t	UD.period.t	H.t	DD.t.i	UD.t.i	…
s002	1	1	0.1491	0.3979	0.2488	0.1069	0.1674	0.0605	…
s002	1	2	0.1111	0.3451	0.234	0.0694	0.1283	0.0589	…
s002	1	3	0.1328	0.2072	0.0744	0.0731	0.1291	0.056	…
s002	1	4	0.1291	0.2515	0.1224	0.1059	0.2495	0.1436	…
s002	1	5	0.1249	0.2317	0.1068	0.0895	0.1676	0.0781	…
s002	1	6	0.1394	0.2343	0.0949	0.0813	0.1299	0.0486	…
s002	1	7	0.1064	0.2069	0.1005	0.0866	0.1368	0.0502	…
s002	1	8	0.0929	0.181	0.0881	0.0818	0.1378	0.056	…

Taking a look at a one-line representation of the data in Table 2. Typing data for Subject 2, Session 1, Repetition 1 are shown in the example. There were several different times that this occurred: 0.1491 seconds (149.1 milliseconds) when the period key was held down; 0.3979 seconds when the period key was released; 0.2488 seconds when the t key was pressed; and so on. "tie5Roanl" is the password that the subjects typed. At the conclusion of the password sequence, the return key is pushed. The sample also includes the key events that the return key produces, namely the return pressed and release. Without logging the shift and caps lock keys' key events separately, the capital letter R is recorded as the key event for the r key. The Comma Separated Values (CSV) file format was used to store the gathered data.

Prior to processing

User-provided raw data must be cleaned, modified, and standardized before being delivered to the classifier. This is known as initial processing, and it makes the data easier to understand. The pre-processing script runs in order to draw out the attributes that are required from all of the unprocessed data. After preprocessing the raw data, 77 users had 155 characteristics produced.
The separation of X and Y coordinates for different typing actions:

$$\sqrt{X^2_{coordinate}} + Y^2_{coordinate} \quad (1)$$

Where Y is the target.
Add X and Y precision values for various typing events:

$$XPrecision_{\text{Down}} + YPrecision_{\text{Down}} \quad (2)$$

when the intended recipient is Y.
The variations in keystroke timestamps and various kinds of operation category incidents originating from attributes with varying initial delays.

$$ActionTimestamp_{Down} - ActionTimestamp_{up,} \quad (3)$$

Where Y is the target.

This process generates several delays, such as:

- Down-Up (du): length of time from hitting and releasing a key
- Up-Down (ud): the period of time from releasing a key and pushing it
- Down-Down (dd): The period of time after hitting two keys in succession
- Up-Up (uu): the length of time between two successive keys being released

The functions rely on the password keys being pressed and the kind of action linked to these different latencies, keystrokes, sizes, X and Y coordinates, and action time stamps.

Features selection

A subset of pertinent features that can be utilized to construct a classification model is chosen through feature selection. When running a classifier for key characteristics, ignoring unnecessary features decreases the amount of data, shortens the model's runtime, and improves performance metrics. Feature selection can lessen classification errors and enhance performance.

Feature extraction phase

Of the initial raw data set, only digestible groupings are left for processing. The Minimum Redundancy Maximum Relevance (mRMR) approach is used to select qualities that exhibit a strong association with the classification variable. Use forward, backward, and floating-point sequential selection to choose subset features. The best features for efficient classification are selected by looking for a subspace of isolated features and utilizing mutual information criteria, such as minimum redundancy and maximum relevance between features. The goal of this feature selection strategy is to increase conditional repeatability, subject to test and validation accuracy. When a compact subset of features is generated from a large number of features, mRMR has been employed as a wrapper technique, which typically results in better classification at lower computing costs. The two variables in the dataset have mutual information provided by:

$$I_{(x,y)} = \iint p(x,y) log \frac{p(x,y)}{p(x)p(y)\,dx\,dy} \quad (4)$$

When the discrete variables x and y are what produce the entropy and their mutual dependence

Training phase

The data collection phase is followed immediately by the training phase. Out of the collected data, 70% was used for training the machine algorithms; the accuracy of the algorithm was evaluated by using the remaining 30% and this was done for each of the four algorithms in this study. In order to make a generalization later on, the training's objective was to teach the algorithm to categorize the subject's keystroke rhythm (Elliot et al., 2019). The first machine learning algorithm to be learned was called Decision Tree. This algorithm's goal is to categorize the properties provided in the desired output. These characteristics, in our example, lead to the user output and constitute the temporal correspondence for specific digraphs. The decision tree is of choice because its output is easy to understand and less effort is required for data preparation during the pre-processing stage. Additionally, the process of building a decision tree is not affected significantly by missed data values during data collection. However, decision trees are notorious in that a little change in the data do cause very big changes in the structure of the decision tree which can lead to instability and takes relatively long time to train the algorithm.

Classification

Decision tree classifier

A decision tree is a mapping of an attribute tree's shape to a desired outcome. Decisions, or path weights between attribute nodes, have an impact on the attributes. The prediction system of the algorithm is enhanced by these decisions, which are variable factors. This algorithm's goal is to categorize the qualities provided in the accurate output. These characteristics, in our example, lead to the user output and constitute the temporal correspondence for specific digraphs. To ensure that every record is accurate, decisions are supplemented in the ideal amount.

Algorithm for Decision Tree (Adapted from: Gurdal and Ibrahim (2018)

```
Input: A collection of practice cases;
Output: A decision tree DT
1.    If the prerequisites for stopping are satisfied, then
2.      make a leaf that represents each of the remaining training case
3.    end
4.    else
5.      using a variety of strategies, choose the optimal characteristic Ax..
6.     identify the current Ax node
7.    Perform for each Vx.j value of attribute Ax.
8.    Add the value Vx.j to an outgoing edge.
9.     Use some of the training cases to iteratively generate a subtree.
10.    that satisfy the 'Ax = Vx.j' requirement
11.   final
12.   final
```

The Random Forest method, which combines several decision trees into a single final output, is the next classifier. The Random Forest Algorithm consists of two stages: the random forest construction stage and the prediction step, which involves using the random forest classifier built in the first stage.

The pseudo-code for Random Forest creation:

1. Given a total of "m" attributes, select "K" features at by chance, where k << m.
2. Applying the ideal split point, identify node "d" among the "K" attributes.
3. Using the optimal split strategy, divide the node into daughter nodes.
4. Steps a through c should be repeated until the "l" number of nodes is reached.
5. To get "n" trees, go through steps a through d "n" times to create a forest.

We use our newly constructed random forest classifier to generate a prediction in the following phase. Below is the random forest forecast's pseudo-code:

1. Forecast the outcome and note the intended outcome (goal) using the test's attributes and the rules of each randomly constructed decision tree.
2. Determine how many people voted for each goal.
3. Assume that the most likely outcome is the final forecast made by the random forest algorithm.

Algorithm for Random Forest (Adapted from: Ahme & Traore, 2013) This algorithm limits overfitting without increasing error due to bias (Liberman, 2016).

1. First, decide how many trees (K) should be generated.
2. For k=1,---,k do.
3. T bootstrap sample from T initialize e=0,t=0,$T_k=\phi$
4. Do until $T_k=N_k$
5. A bag of covariance represented by vector Ck is generated.
6. Using your preferred decision tree technique, create Tree h (I, Ck).
7. Each Tree computes its estimation using a single matrix from the set of covariance matrices at I.
8. Votes are cast for the most popular covariance matrix at picture I by each tree.
9. The most-voted-for matrix among h1, h2,....., hk is chosen as the popular covariance matrix at I.
10. $h_i = \text{argmax}_y \sum_{k=1}^{k} I(h_k(x) = y)$
11. Return a hypothesis hi
12. End for

Support vector classifiers

Support Vector Machine (SVM) learning was the third machine learning technique employed in this study. It performs better memory management than many other learning algorithms, but it has trouble with very large datasets because to the long training period. Using this approach, data is graphed, and the ideal hyper plane with the largest margin between data segments is produced. Because of this, a prediction can be made based on the category to which the fresh set of data belongs; the classes are represented by the groups on either side of the hyperplane. The forecast is more confidently made the higher the margin distance with the hyper plane. Kernel methods like SVM-Radial Basis Function (RBF), which controls information from greater dimensions in order to produce a hyper plane, allow a line that resembles a circle to be drawn around a class to generate the maximum margin whenever classes cannot be differentiated linearly. Generally, SVM hyper planes are linear separations of the classes.

K-nearest neighbors classifier

The K-nearest Neighbors machine learning algorithm is the last one. K-nearest Neighbors has a brief training phase, is highly accurate, and is simple to grasp. However, the KNN approach requires a lot of memory and requires expensive computations when employed with huge datasets. Accuracy is further complicated by the fact that data quality is important Mariana (2020). To estimate a new data point's class or continuous value, K Nearest Neighbors (data points) are considered. Euclidean distance is used in the KNN algorithm to calculate the separation between two points. This is how Euclidean distance is calculated as depicted in equation 1.

$$d(p,q) = \sqrt{\sum_{i=1}^{n} (qi - pi)2} \quad (5)$$

Algorithm for K Nearest Neighbors

1. Choose the number K of the neighbors
2. Calculate the number of neighbors' distances using Euclidean geometry.
3. Choose the K nearest neighbors based on the assessed Euclidean distance.
4. Ascertain the number of information points each of these k neighbors has in every group.
5. Place the additional information points in the group of data with a significant neighbor count.
6. End

.

Testing

The trained algorithms were tested to determine whether data belonged to a specific user using the remaining 30% of the data. The algorithms were put to five different tests.

Comparison of algorithms

This step's major objective is to evaluate and compare how well the algorithms handled the test dataset. The dataset includes keyboard data from 51 people that were collected over the course of 8 sessions over the course of 6 months; each session recorded 50 samples for a total of 400 samples per user. By doing this, we aim to evaluate how the amount of data provided influences the accuracy of all four techniques.

Model Implementation

The flowchart shown in Figure 2. Represents the overall process involved in the proposed system.

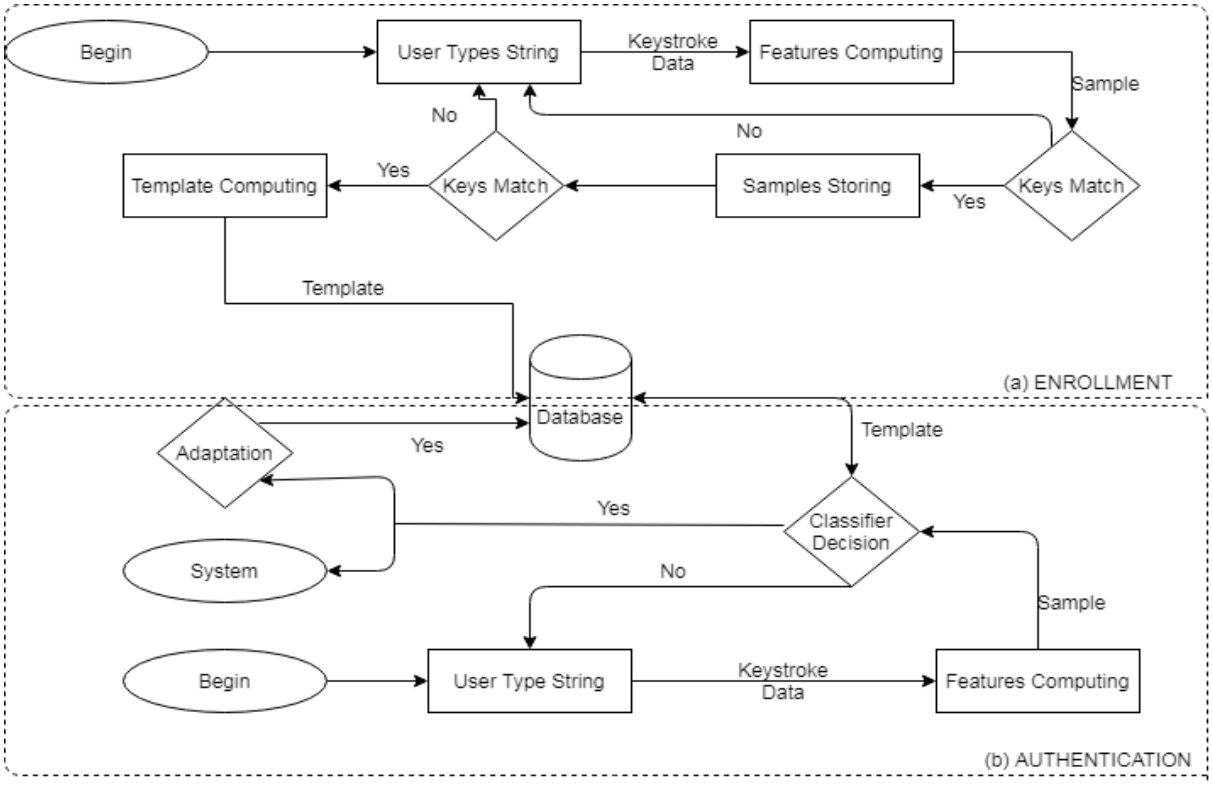

Figure 2: Flowchart for Keyboard dynamics

There are often two fundamental parts to this as depicted in figure 2. The first section discusses enrolment. A new user must register with the keyboard dynamics system. New users must enter their login information and

password. Once the registration process is successfully finished, the new user needs to provide their username and password to save their typing habits and reference template in the server's database. Regular users can access the second part of the Keyboard Dynamics system by entering their login credentials if they have already registered. The client will record and send keystroke information to the server. This verification phase can only be completed if the user's keyboard dynamics information matches the reference template for their username and password. This method makes it possible to distinguish between a legitimate user and a fake user.

Classification report

The performance metrics evaluated on the dataset are:

1. 1. Often used to show how well a performance classification algorithm performs on a sample data set where the real values are determined, the Confidence Matrix is a tabular depiction.
2. Accuracy: is the proportion of all data points that were accurately anticipated.

$$\Pr = \frac{TP+TN}{TP+TN+FP+FN} \qquad (6)$$

3. Recall Score: (also referred to as sensitivity) is the percentage of pertinent cases that were found.

$$\text{Re} = \frac{TP}{TP+FN} \qquad (7)$$

4. False Positive Rate: is the proportion of all negative forecasts divided by the number of negative predictions that turned out to be positive.

$$\text{False Positive Rate} = \frac{TN}{TN+FP} \qquad (8)$$

5. Precision Score: (more commonly known as the positive predictive value) is the percentage of pertinent cases found in the retrieved instances.

RESULTS AND DISCUSSION

For every user, a positive and negative case was made, with 0 for the real user and 1 for the impostor. Scaling the features and dividing a new data shape of (38000, 35) into 70-30 train test splits were done. Putting the four models into practice with GridSearchCV, which uses cross-validation to ensure that the mode is not overfitting the dataset and thoroughly tests each parameter? The models' classification reports are displayed in the tables in tabs 2 through 5 below.

1. Support Vector Machine (SVM)

Table 2: SVM Result

Classification Report	Precision	Recall	F1-score	Support
0.0	0.96	0.98	0.97	6106
1.0	0.98		0.97	5294
Accuracy			0.97	11400
Macro avg	0.97	0.97	0.97	11400
Weighted avg	0.97	0.97	0.98	11400

With an EER of 0.029 and an accuracy of 97.55%, SVM has done fairly well. Kennel: According to the F1-score, the optimal parameter for the dataset is "rbf." The SVM's ROC curve is shown in Figure 3, with the y-axis representing the true positive rate and the x-axis representing the false positive rate.

Figure 3: ROC curve of SVM

2. KNN

Table 3: KNN Result

Classification Report				
	Precision	Recall	F1-score	Support
0.0	0.96	0.99	0.98	6119
	0.99	0.96	0.97	5281
accuracy			0.97	11400
Macro avg	0.97	0.97	0.97	11400
Weighted avg	0.97	0.97	0.97	11400

KNN has a lower accuracy of 97.28% than SVM model. But EER is higher for the KNN with 0.043; therefore SVM is a better model than KNN for the dataset. The best parameters for comparing parameters performance in GridSearch is 'n_neighbors': 3. Figure 4 depicts the ROC curve for KNN where the x-axis showing False Positive Rate and y-axis showing the truh positive rate.

Figure 4: ROC curve for KNN

3. Random Forest

Table 4: Random Forest Result

Classification Report				
	Precision	Recall	F1-score	Support
0.0	0.97	0.99	0.98	6119
1.0	0.98		0.98	5281

Accuracy			0.98	11400
Macro avg	0.98	0.98	0.98	11400
Weighted avg	0.98	0.98	0.98	11400

Random Forest has the higher accuracy of 98% than SVM, KNN and Decision Tree model. But EER is lower for the Random Forest with 0.0401; therefore, SVM is a better model than Random Forest for the dataset. The best parameters for comparing parameters performance in GridSearch is 'criterion': 'entropy', 'max_depth':8, 'max_features': 'auto', 'n_estimators': 200. Figure 5 depicts the ROC curve for Random Forest where the x-axis showing False Positive Rate and y-axis showing True Positive Rate.

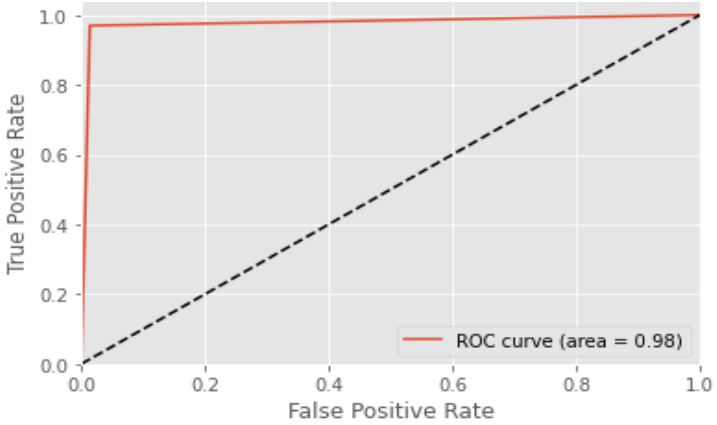

Figure 5: ROC Curve for Random Forest

4. Decision Tree

Table 5: Decision Tree Result

Classification Report				
	Precision	Recall	F1-score	Support
0.0	0.93	0.96	0.95	6074
1.0	0.95	0.92	0.94	5326
accuracy			0.94	11400
Macro avg	0.94	0.94	0.94	11400
Weighted avg	0.94	0.94	0.94	11400

Decision Tree has the lower accuracy of 94.26% than the other three models. But EER is higher than Random Forest; SVM & Decision Tree with 0.075, therefore SVM, Random Forest & KNN is a better model than Decision tree for the dataset. The best parameters for comparing parameters performance in GridSearch is 'criterion': 'entropy', 'max_depth':8, 'max_features': 'auto'. Figure 6 depicts ROC curve of Decision Tree where the x-axis showing False Positive Rate and y-axis showing True Positive Rate

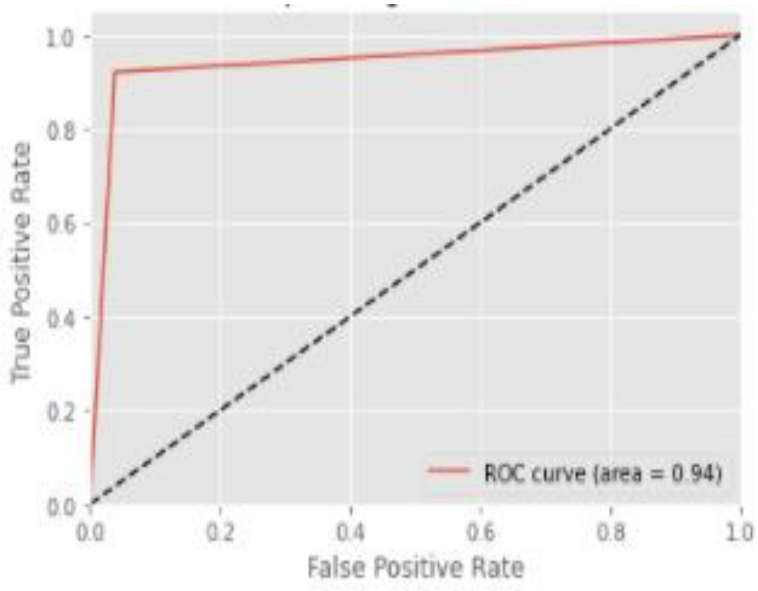

Figure 6: ROC curve of Decision Tree

EVALUATION

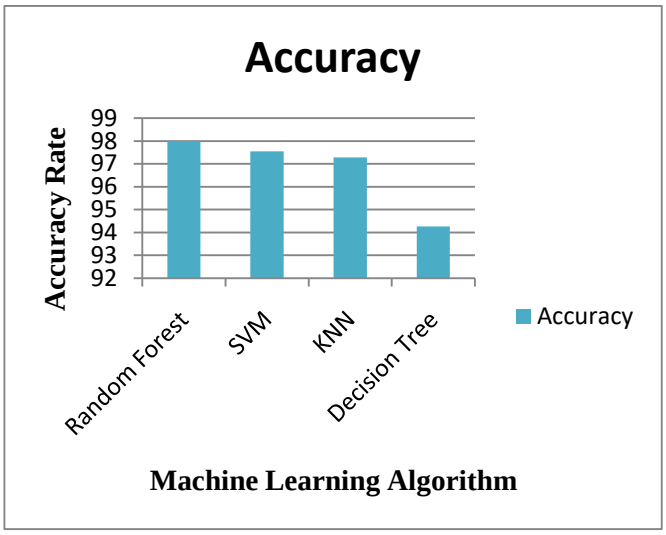

Figure 7: Comparison of Model Accuracy

Figure 7 above depicts the accuracy rate of the classifier, with Random Forest providing the highest accuracy with 98%, SVM providing an accuracy rate of 97.55% and performance that is relatively poor compared to random forest, KNN providing a performance rate that is close to random forest at 97.28%, and Decision Tree providing the lowest accuracy rate with 94.26% and performance that is very poor for the dataset.

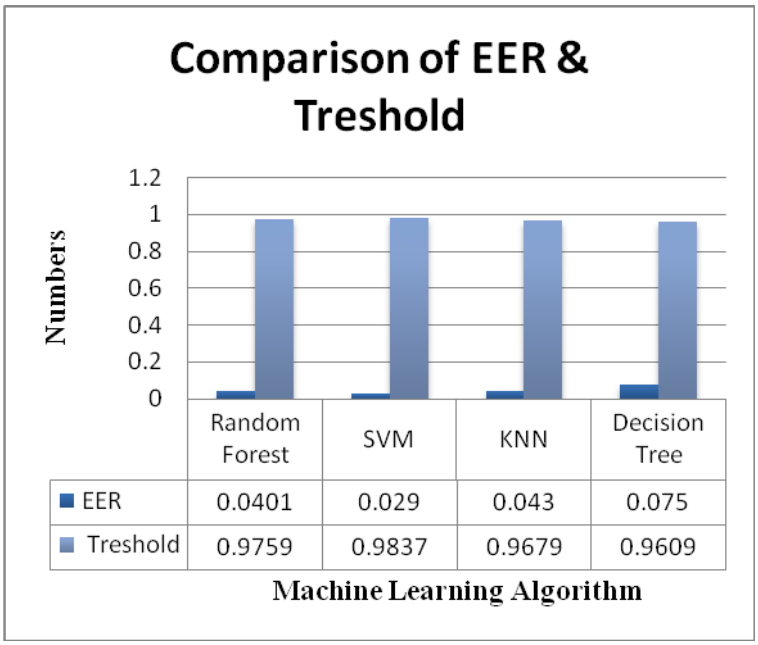

Figure 8: Comparison of EER and Threshold

Figure 8 shows the comparison of equal error rate and threshold where SVM gave the lower EER of 0.029 which is more accurate than Random Forest which gave a EER of 0.0401, KNN which gave a EER of 0.043, and Decision Tree gave a EER of 0.075. The accuracy of the biometric system increases with a reduced equal error rate.

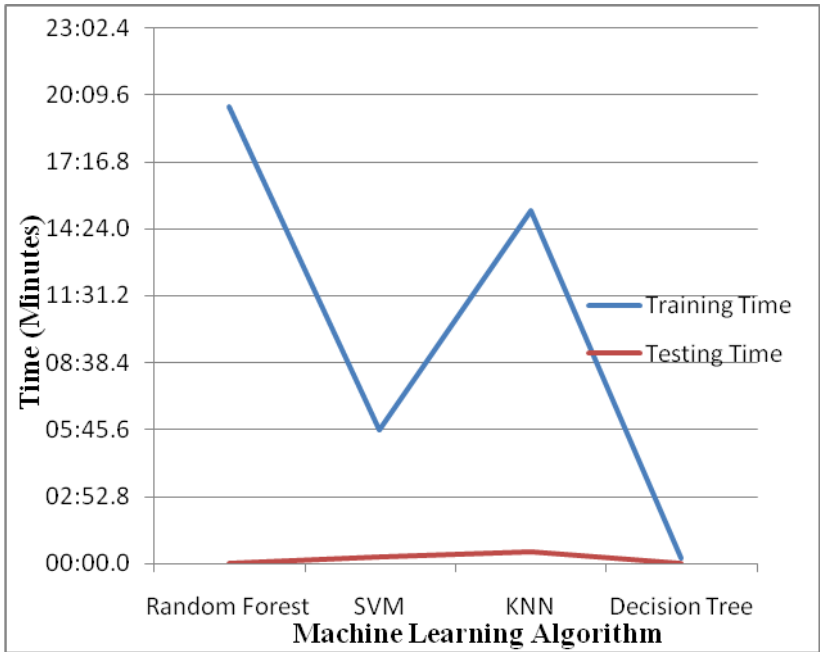

Figure 9: Comparison of Training & Testing time

In figure 9 above, Random Forest required 19 minutes to train the data, followed by KNN at 15 minutes, however these times pale in comparison to SVM's 5 minutes and Decision Tree's 14 seconds. Classifiers' testing time was brief.

CONCLUSION AND RECOMMENDATION

The best accuracy results for the full dataset after classification using several classifiers are displayed in Tables 2 through Table 5. In general, it was found that when the performance of all the classifiers employed in the experiment was compared, Decision Tree performed the worst, K-nearest Neighbors and Support Vector Machine performed fairly similarly, while SVM performed better, and Random Forest performed exceptionally. For this issue, random forest can be relied upon. It is possible to authenticate keyboard dynamics by using Random Forest as the system model. It distinguishes between a legitimate user and a fraud. The Random Forest classifier generated the best accuracy values across all datasets. In conclusion, K-nearest Neighbors & Support Vector outperformed Decision Tree when all the classifiers used in the experiment were compared. The Random Forest classifier performs well for this comparative analysis and can be used for all multi-class classification problems.

ACKNOWLEDGMENT

The authors wish to acknowledge Prof. Olusegun Folorunsho of the Department of Computer Science, Federal University of Agriculture Abeokuta for his suggestions that made this work a success.

REFERENCES

Ahmed, A.A. **&** Traore, I. (2013). Biometric recognition based on free-text keystroke dynamics. *IEEE transactions on cybernetics*, 44(4), 458-472.

Andrean, A., Jayabalan, M., & Thiruchelvam, V. (2020). Keystroke dynamics based user authentication using deep multilayer perceptron. *International Journal of Machine Learning and Computing*, 10(1),134-139.

Antal, M. & Nemes, L. (2016). The MOBIKEY Keyboard dynamics Password Database: Benchmark Results, pages 35-46. Springer International Publishing, Cham

Alsuhibany, S.A., Almushyti, M. & Alkhudhayr, F. (2020). Investigating the accuracy of free-text keyboard dynamics authentication in touchscreen devices. *International Journal of Biometrics*, 12(4), 430. https://doi.org/10.1504/ijbm.2020.110816

Arulkumaran, K., Deisenroth, M. P., Brundage, M., & Bharath, A. A. (2017). Deep reinforcement learning: A brief survey. *IEEE Signal Processing Magazine*, 34(6), 26-38.

Auyporn, W. & Vongpradhip, S. (2015). A robust image encryption method based on bit plane decomposition and multiple chaotic maps. *Int. J. Signal Process. Syst*, 3(1),

Banerjee, S. P. & Woodard, D. (2012). Biometric Authentication and Identification Using Keyboard dynamics: A Survey. *Journal of Pattern Recognition Research*, *7*(1), 116–139. https://doi.org/10.13176/11.427

Baynath, P., SunjivSoyjaudah, K. M. &Heenaye-Mamode Khan, M. (2018). Machine Learning Algorithm on Keyboard dynamics Pattern. *2018 IEEE Conference on Systems, Process and Control (ICSPC)*. Published.

BhadriNaarayanan P. (2020). Comparing the Performance of Anomaly Detection Algorithms. *International Journal of Engineering Research And*, *V9*(07). http s://doi.org /10.17577/ij ertv9 I s 070532. https://doi.org/10.1109/spc.2018.8704135

Bell, J. (2022). What is machine learning? *Machine Learning and the City: Applications in Architecture and Urban Design*, 207-216.

Bhatnagar, M., Jain, R. K., &Khairnar, N. S. (2013). A survey on behavioral biometric techniques: mouse vs keyboard dynamics. *Int. J. Comput. Appl*, *975*, 8887.

De-Marcos, L., Martínez-Herráiz, J. J., Junquera-Sánchez, J., Cilleruelo, C. & Pages-Arevalo, C. (2021). Comparing machine learning classifiers for continuous authentication on mobile devices by keystroke dynamics. *Electronics*, 10(14), 1622.

Chang, H. C., Li, J., Wu, C. S. & Stamp, M. (2022). Machine learning and deep learning for fixed-text keystroke dynamics. In *Artificial Intelligence for Cybersecurity* (pp. 309-329).

Dorca-Josa, A. (2017). *Identifying users using Keyboard dynamics and contextual information* (Doctoral dissertation, Universitatd'Andorra).

Draffin, B., Zhu, J. & Zhang, J. (2014). Keysens: Passive user authentication through micro-behavior modeling of soft keyboard interaction. In *Mobile Computing, Applications, and Services: 5th International Conference, MobiCASE 2013, Paris, France, November 7-8, 2013, Revised Selected Papers 5* (pp. 184-201). Springer International Publishing.

Elliot, K., Graham, J., Yassin, Y., Ward, T.,Caldwell, J. & Attie, T. (2019). A comparison of machine learning algorithms in keystroke dynamics. In *2019 international conference on computational science and computational intelligence (CSCI)* (pp. 127-132). IEEE.

Eude, T. & Chang, C. (2017). One class SVM for biometric authentication by keyboard dynamics for remote evaluation. *Computational Intelligence*, *34*(1), 145–160. https://doi.org/10.1111/coin.12122

Giot, R., El-Abed, M. & Rosenberger, C. (2009). Greyc keystroke: a benchmark for keystroke dynamics biometric systems. In *2009 IEEE 3rd International Conference on Biometrics: Theory, Applications, and Systems* (pp. 1-6). IEEE.

Goldberg, I. & Wagner, D. (1996). Randomness and the Netscape browser. *Dr Dobb's Journal-Software Tools for the Professional Programmer*, *21*(1), 66-71.

Gurdal, M. M. & Gebze, I. S. (2018). A novel method to improve performance of the keystroke dynamics based user authentication. In *Proceedings of the International Conference on Innovations in Computing***.** (ICIC 2018): 5-8.

Huang, J., Hou, D., Schuckers, S., Law, T. & Sherwin, A. (2017). Benchmarking keystroke authentication algorithms. In *2017 IEEE Workshop on Information Forensics and Security (WIFS)*(pp. 1-6). IEEE.

Juola, P., Noecker, J. I., Stolerman, A., Ryan, M. V., Brennan, P. &Greenstadt, R. (2013). Keyboard-behavior-based authentication. *IT Professional*, *15*(4), 8-11

Kim, D. I., Lee, S. & Shin, J. S. (2020). A new feature scoring method in keystroke dynamics-based user authentications. *IEEE Access*, *8*, 27901-27914..

Kim, J., Kim, H. & Kang, P. (2018). Keyboard dynamics-based user authentication using freely typed text based on user-adaptive feature extraction and novelty detection. *Applied Soft Computing*, *62*, 1077–1087. https://doi.org/10.1016/j.asoc.2017.09.045

Kim, J. & Kang, P. (2020). Freely typed keyboard dynamics-based user authentication for mobile devices based on features. *Pattern Recognition*, *108*, 107556. https://doi.org/10.1016/j.patcog.2020.107556

Krishnamoorthy, S., Rueda, L., Saad, S. &Elmiligi, H. (2018). Identification of User Behavioral Biometrics for Authentication Using Keyboard dynamics and Machine Learning. Proceedings of the 2018 2nd International Conference on Biometric Engineering and Applications -ICBEA 18. doi:10.1145/3230820.3230829

Liberman, N. (2017). Decision Trees and Random Forests–Towards Data Science.

Maiorana, E., Kalita, H. &Campisi, P. (2020). Mobile keyboard dynamics for biometric recognition: An overview. *IET Biometrics*, 10(1), 1–23. https://doi.org/10.1049/bme2.12003

Mariana Chatterjee (2020). A quick introduction to KNN algorithm. [Available: https://www.mygreatlearning.com/blog/knn-algorithm-introduction/]

Mondal, S. & Bours, P. (2016, February). Combining keystroke and mouse dynamics for continuous user authentication and identification. In *2016 IEEE international conference on identity, security and behavior analysis (ISBA)* (pp. 1-8). IEEE.

Pavaday, N. &Soyjaudah, K. M. S. (2007). Performance of the K Nearest Neighbor in Keyboard Dynamic Authentication. In *Proceedings of the 2007 Computer Science and IT Education Conference* (pp. 599-604).

Piugie, Y. B. W., Di Manno, J., Rosenberger, C. & Charrier, C. (2022) Keystroke dynamics based user authentication using deep learning neural networks. In *2022 International Conference on Cyberworlds (CW)* (pp. 220-227). IEEE.

Revett, K., Gorunescu, F., Gorunescu, M., Ene, M., Magalhaes, S. & Santos, H. (2007). A machine learning approach to keystroke dynamics based user authentication. *International Journal of Electronic Security and Digital Forensics*, *1*(1), 55-70.

Ryu, R., Yeom, S., Kim, S. H. & Herbert, D. (2021). Continuous multimodal biometric authentication schemes: a systematic review. *IEEE Access*, *9*, 34541-34557.

Sharma, A., Jureček, M. & Stamp, M. (2023). Keystroke Dynamics for User Identification. *arXiv preprint arXiv:2307.05529*.

Uschold, M. & Gruninger, M. (2004). Ontologies and semantics for seamless connectivity. *ACM SIGMod Record*, *33*(4), 58-64.