Cite this research:
Nova, K.,(2022). Analyzing Keystroke Dynamics for User Authentication: A Comparative Study of Feature Extraction and Machine Learning Models SSRAML SageScience, 5(2), 67–80.

Article history:
Received:
April/12/2021
Accepted:
November/08/2022
Published:
November/20/2022

Analyzing Keystroke Dynamics for User Authentication: A Comparative Study of Feature Extractions and Machine Learning Models

Kannan Nova

Abstract

Keystroke dynamics, the analysis of typing patterns, has gained considerable attention as a biometric authentication method due to its potential for non-intrusive and continuous user identification. In this research, we investigate the effectiveness of different feature extraction techniques and machine learning models in analyzing keystroke data for user authentication. The dataset used in this study consists of 50 users' attempts to enter the 11-character string 'Restaurant' seven times, with recorded timestamps for each key press and release corresponding to the initial key press. To explore the potential of keystroke dynamics for user authentication, we first extract a range of features from the dataset. These features include raw features such as key-press time and key-release time, as well as derived features such as durations, latencies, digraphs, and second-order statistical measures. We then evaluate the performance of three popular machine learning models: Decision Tree, Random Forest, and XGBoost, using these features. Our results indicate that the inclusion of first and second-order features significantly enhances the performance of the machine learning models compared to raw features alone. The Random Forest and XGBoost models demonstrate superior performance across all feature combinations, achieving high R-squared values and low root mean square error (RMSE) values. This highlights the importance of capturing both local and higher-level patterns in keystroke data for accurate user authentication.

Keywords: Authentication, Biometric, Feature extraction, Keystroke dynamics, Machine learning, User identification, User authentication

Introduction

Behavioral biometrics refer to a collection of distinctive innate behaviors exhibited by individuals, which can be utilized as a means of user identification [1]. One prominent example of behavioral biometrics is keystroke dynamics, which involves analyzing the typing patterns of an individual. Keystroke dynamics take into account various factors such as the duration of key presses, the time intervals between keystrokes, and the pressure applied while typing [2]. By capturing and analyzing these unique patterns, keystroke dynamics can serve as an effective method for user authentication and verification [3], [4].

Apart from keystroke dynamics, there exists a wide array of behavioral biometrics that can be employed to identify users. Some of these biometric measures involve physical actions, such as the gait or stride of an individual. Gait analysis involves studying the manner in which a person walks, including factors like stride length, pace, and posture. Similarly, lip movement analysis focuses on the unique patterns of lip motion exhibited by individuals

while speaking [5]. These physical behavioral biometrics can be utilized to enhance security systems by providing an additional layer of user authentication.

In addition to physical actions, technological-based behavioral biometrics can also be leveraged for user identification. Graphical User Interface (GUI) interaction is one such example, which involves analyzing how individuals interact with various elements of a graphical user interface, such as buttons, sliders, and menus. By examining the patterns and preferences in GUI interaction, it becomes possible to differentiate between different users. Another example is game strategy analysis, which entails studying the individual approaches and decision-making patterns exhibited by users in gaming environments. By understanding the unique strategies employed by players, game strategy analysis can aid in user identification and fraud detection [6].

Keystroke dynamics, as a form of behavioral biometric, focus on capturing and analyzing the unique typing patterns exhibited by individuals [7]. The origins of keystroke dynamics can be traced back to the 1800s, where Morse Code operators could be distinguished from one another based on their distinct use of the device. This early recognition of individual typing patterns laid the foundation for the development of keystroke dynamics as a reliable method of identification and authentication.

One of the notable advantages of keystroke dynamics is its inherent reliability. Unlike traditional authentication methods such as passwords or PINs, which can be forgotten or compromised, keystroke dynamics are based on inherent behaviors that are learned slowly over time through repetition and consistency [8]. These characteristics make keystroke dynamics highly desirable for identification and authentication purposes, as they offer a level of reliability that is not easily replicated.

When capturing keystroke data, there are three key metrics that are typically recorded. Firstly, the keycode refers to the specific key that is pressed by the user. This information is essential for understanding the individual's typing behavior and the unique patterns associated with each key. Secondly, the timestamp of keypress records the precise moment when a key is initially pressed down by the user. This temporal data provides insights into the timing aspects of the typing pattern. Lastly, the timestamp of key release captures the moment when the user releases the pressed key [9]. This metric helps to determine the duration of each keystroke and the timing intervals between consecutive keystrokes.

By recording and analyzing these key metrics, keystroke dynamics can serve as a robust biometric modality for user identification and authentication. The unique typing patterns and timing characteristics of individuals can be effectively used to distinguish one user from another, enhancing security systems and providing reliable means of access control. As technology advances, keystroke dynamics are likely to continue evolving, becoming even more accurate and widespread in various applications that require secure and reliable user identification.

The capture phase plays a crucial role in the overall biometric authentication process, as it involves collecting data from individuals to build their unique biometric models. The capture process occurs at two significant stages: enrollment and verification.

During the enrollment phase, multiple samples of the user's biometric data are collected in order to construct their individual model. The exact procedure for enrollment can vary

depending on the type of keystroke dynamics system being used. For instance, some systems may require users to type a fixed string multiple times, while others may monitor the user's computer usage to capture a comprehensive dataset. Additionally, the quantity of data required for enrollment can vary significantly across different studies and implementations. Some studies may require a large amount of data to establish a robust biometric model, while others may achieve satisfactory results with a smaller dataset.

In the verification phase, a single sample is collected from the user for comparison with their pre-established biometric model. From this sample, various features are extracted to create a representation of the user's unique typing pattern. These features may include metrics such as key press duration, key release timing, and pressure applied while typing. Once the features are extracted, they are compared to the stored biometric model of the claimant [10]. The comparison process involves measuring the similarity between the extracted features and the stored model, usually through mathematical algorithms or statistical techniques. Based on this comparison, a decision is made regarding the authenticity of the user's identity.

Effective capture during both enrollment and verification phases is crucial for the accuracy and reliability of biometric authentication systems. In the enrollment phase, capturing a diverse range of user samples ensures that the biometric model accurately represents the individual's unique typing pattern [11]. Adequate sample size and variation in the data contribute to a more robust and reliable model. During verification, the quality of the captured sample plays a vital role in determining the accuracy of the authentication process. Factors such as sensor precision, environmental conditions, and user cooperation can influence the quality of the captured sample and, consequently, the overall performance of the system.

Feature extraction and data preprocessing

Keystroke dynamics analysis involves the extraction and analysis of various features derived from recorded keystroke data [12].

Raw features

Recorded keystroke data, namely, key-press time and key-release time for each key.

First order features:

The most commonly extracted characteristics are local or first order features, which are calculated by subtracting timing values. Duration refers to the length of time a key is pressed. For a given key "i", it is determined using the following formula [12]:

duration = time when event = RELEASE - time when event = PRESS

We then obtain a timing vector, also known as PR in literature, which contains the duration of each key press in the order they were pressed. For all "i" where $1 \leq i \leq n$, PR(i) represents the duration of the "i-th" key press.

There are different types of latencies that can be utilized, which are computed by calculating the time differences between two key events.

PP Latency: This refers to the time difference between the pressing of each key. It is obtained using the following equation [12]:

For all "i" where 1 ≤ i < n, PP(i) = time when (i+1)-th event = PRESS - time when i-th event = PRESS

RR Latency: This represents the time difference between the release of each key. It is calculated using the following equation:

For all "i" where 1 ≤ i < n, RR(i) = time when (i+1)-th event = RELEASE - time when i-th event = RELEASE

RP Latency: This indicates the time difference between the release of one key and the pressing of the next key. It can be obtained using the following equation:

For all "i" where 1 ≤ i < n, RP(i) = time when (i+1)-th event = PRESS - time when i-th event = RELEASE

Another frequently encountered concept in the literature is the notion of a digraph. A digraph represents the time required to press two keys sequentially. The digraph features D of a password are computed as follows:

For all "i" where 1 ≤ i < n, Di = time when (i+1)-th event = RELEASE - time when i-th event = PRESS.

Second order features:

Some characteristics are not directly extracted from the raw biometric data, but rather derived from the first order features.

Minimum/maximum: This involves determining the minimum and maximum values for each type of data (latency and duration).

Mean/standard deviation: This entails calculating the average value and the standard deviation for each type of data (latency and duration).

Slope: By examining the slope of the biometric sample, we are interested in the overall pattern of typing. We expect that users maintain a consistent typing style even if their speed may vary. The new set of features is computed as follows:

For all "i" where 1 ≤ i < n, result(i) = source(i+1) - source(i) (8)

Spectral information: A discrete wavelet transformation can be applied to the originally extracted features. All the operations are performed on the data that has undergone wavelet transformation.

Figure 1. Correlation heatmap among features

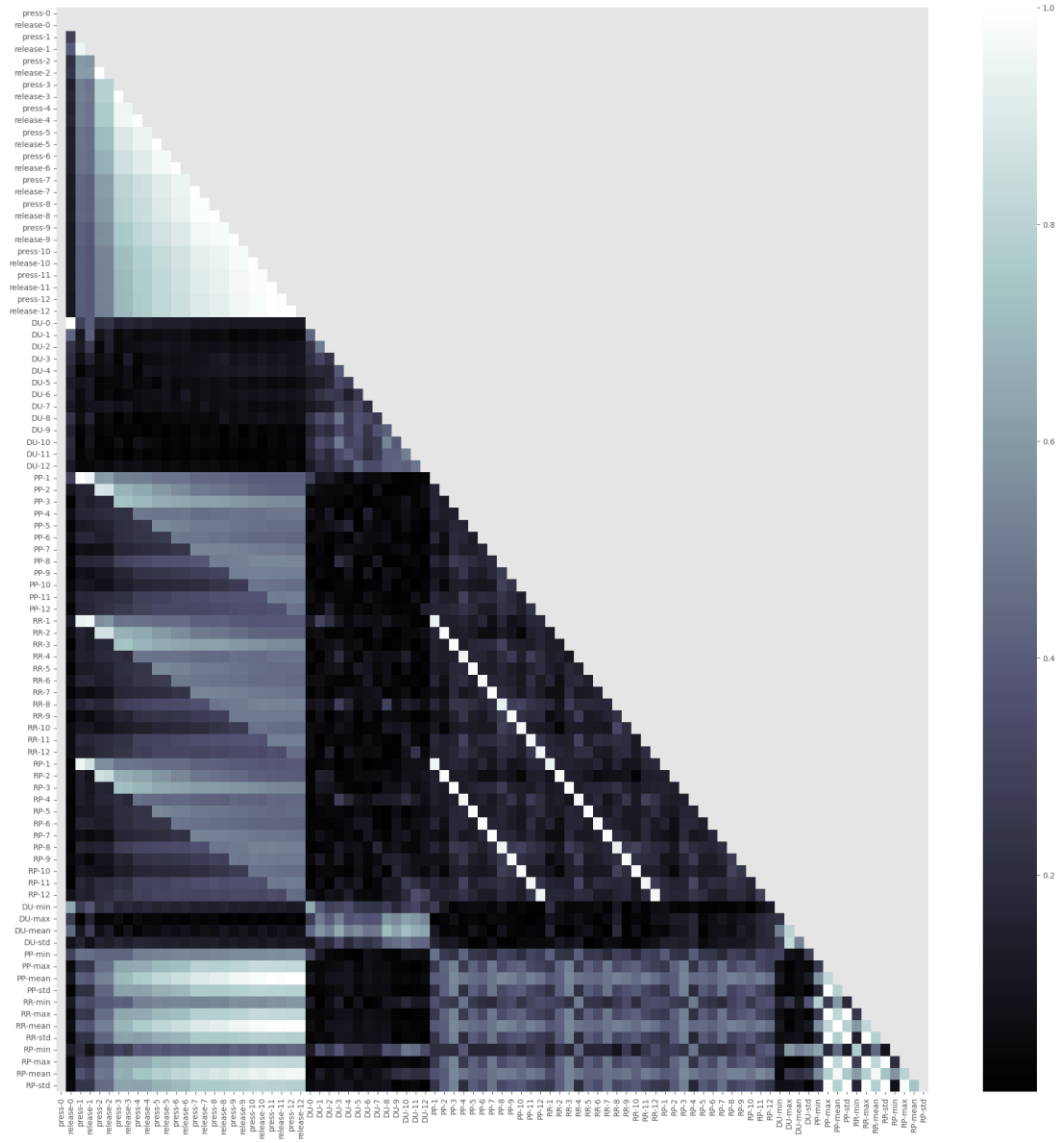


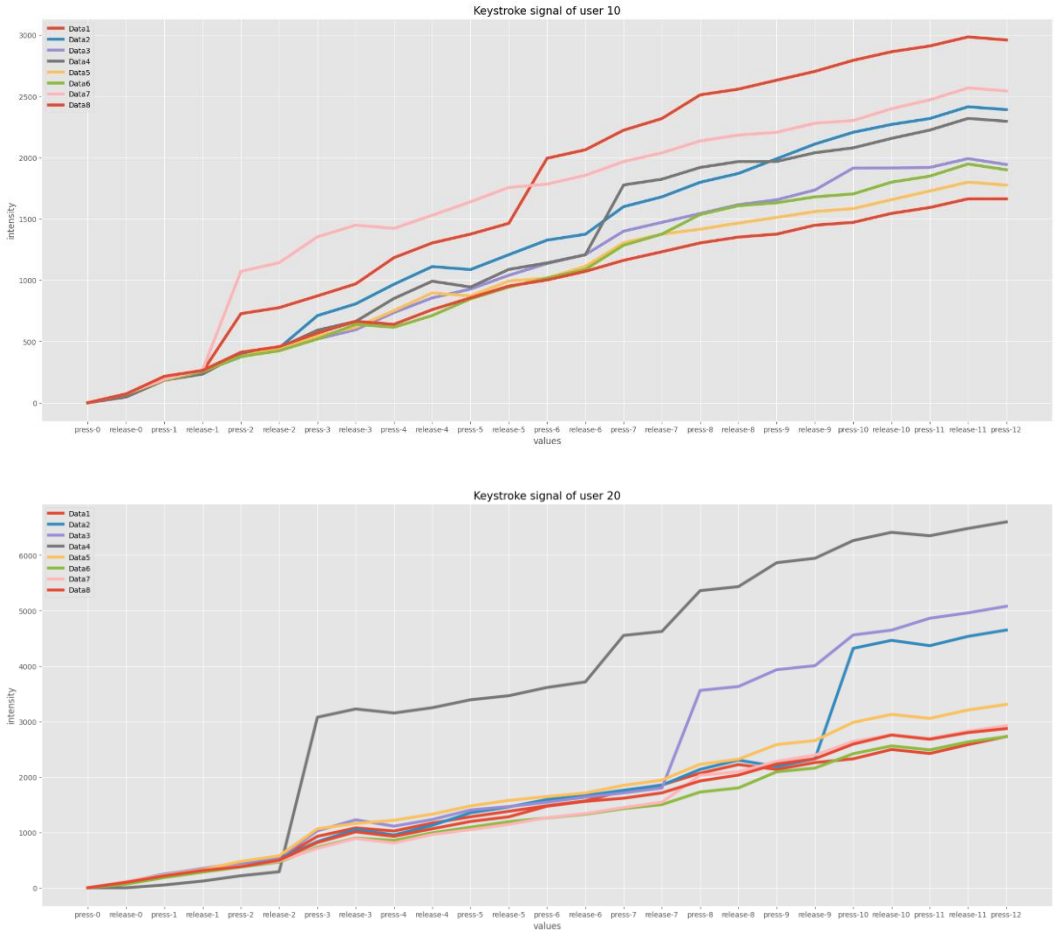
Table 1. First few correlations among features

Features A	Features B	Correlation score
press-12	release-12	0.999361
PP-mean	release-12	0.999361
RR-mean	release-12	0.999228
release-7	press-7	0.998849
release-11	press-11	0.998824
release-9	press-9	0.99882
PP-mean	RR-mean	0.998753

Figure 1 shows the correlation heatmap among features. The provided data in table 1 shows a set of correlation scores. The "press-12" and "release-12" have a correlation score of 0.999361, suggesting a very strong positive correlation between them. Similarly, in the second row, "PP-mean" and "release-12" also have a correlation score of 0.999361, indicating a strong positive correlation between these two features. "RR-mean" and "release-12" have a correlation score of 0.999228, which is also very high. This indicates a strong positive correlation between these features, meaning that they tend to vary together in a similar pattern.

We also observe high correlation scores between "release-7" and "press-7," "release-11" and "press-11," and "release-9" and "press-9," respectively. All three pairs have scores above 0.9988, indicating a strong positive correlation. This suggests that when one feature is released, the corresponding feature is likely to be pressed. "PP-mean" and "RR-mean" have a correlation score of 0.998753, indicating a strong positive correlation.

Figure 2. Keystrokes behavior for 2 randomly selected users



To preprocess our continuous data and convert it into discrete values, we utilized a technique called binning or discretization. This process involves dividing the range of

continuous feature values into distinct intervals or bins and assigning discrete labels to each bin. By doing so, we can simplify the representation of our data and make it more suitable for certain algorithms. Keystroke behaviors of two randomly selected (user 10, and user 20) are shown in figure 2.

For the binning process, we employed a specific algorithm called KBinsDiscretizer. This algorithm is designed to perform the discretization task effectively. It takes the continuous features as input and outputs discrete values that fall within the range of [0, numBins), where numBins represents the number of bins or intervals we want to create.

Once we obtained the discrete representations of our features using KBinsDiscretizer, we proceeded to split our data into two sets: the training dataset and the validation dataset. This division is crucial for evaluating the performance of classifiers or predictive models. To maintain the integrity of the data distribution while splitting, we employed stratified sampling. This sampling technique ensures that the class distribution in both the training and validation datasets is representative of the original dataset. In our case, we allocated 80% of the data to the training dataset and the remaining 20% to the validation dataset. This division allows us to train our classifiers on a substantial portion of the data while reserving a separate subset for assessing their performance.

Classification methods

Decision trees are used to make predictions and decisions by creating a hierarchical structure of rules and conditions [13], [14]. They work by partitioning the input data based on various features and attributes, recursively splitting the dataset into smaller subsets at each node. These splits are determined by evaluating the significance of different features in terms of their ability to separate the data and make accurate predictions. The decision tree algorithm aims to maximize the information gain at each step, selecting the most informative feature to split the data [15]. This process continues until a termination condition is met, such as reaching a predefined depth or achieving a minimum number of samples in each leaf node. Decision trees excel at handling both numerical and categorical data, providing interpretable and explainable models. However, they are prone to overfitting when the tree becomes too deep or when the dataset contains noisy or irrelevant features [16], [17].

Random forests, on the other hand, are a powerful ensemble method that leverages decision trees to improve prediction accuracy and reduce overfitting. In random forests, multiple decision trees are built independently using different subsets of the training data and a random selection of features. Each tree in the forest is constructed by sampling the training data with replacement, a process known as bootstrap aggregating or bagging. By creating a diverse set of trees that are trained on different subsets of data, random forests promote robustness and generalize well to unseen examples [18]. Additionally, at each split in a decision tree, only a subset of randomly chosen features is considered, further adding to the diversity of the forest [19]. The final prediction of a random forest is obtained by aggregating the predictions of individual trees, either through majority voting in classification tasks or averaging in regression problems [15].

XGBoost, short for Extreme Gradient Boosting, is a highly sophisticated and powerful machine learning algorithm that belongs to the family of gradient boosting methods [20], [21]. It excels in solving complex prediction and regression problems by creating an

ensemble of weak learners, typically decision trees, and iteratively optimizing their performance. XGBoost combines the strengths of gradient boosting with several innovative techniques to achieve exceptional accuracy and efficiency. At its core, XGBoost utilizes gradient descent to iteratively train a sequence of decision trees. Each subsequent tree is built to correct the errors or residuals made by the previous trees, with a focus on minimizing a specific loss function. The algorithm assigns higher weights to the misclassified samples, enabling subsequent trees to prioritize their correct classification. XGBoost provides flexible options for different loss functions, allowing users to tailor the algorithm to specific tasks such as binary classification, multiclass classification, and regression [22], [23].

R-square measures the proportion of the total variance in the dependent variable that can be explained by the independent variables in the model The R-square value is calculated by dividing the sum of squares of the regression (explained variance) by the total sum of squares (total variance) . A higher R-square value suggests a better fit of the model to the data. Root Mean Squared Error (RMSE) is a commonly used evaluation metric in regression analysis. It measures the average magnitude of the residuals or errors between the predicted values and the actual values in a regression model. RMSE provides a measure of the model's accuracy in predicting the dependent variable. To calculate RMSE, the squared differences between the predicted and actual values are averaged, and then the square root is taken to obtain the final value. RMSE is preferred over mean absolute error (MAE) because it penalizes larger errors more heavily, making it more sensitive to outliers. A lower RMSE value indicates better predictive performance, as it reflects smaller prediction errors.

Results and discussion

We calculated R-square and RMSE to evaluate The Decision Tree, Random Forest, and XGBoost. The first, second, third, and fourth stages of of our calculations involved selecting raw features, first order features, second order features, and combination of 1st and 2nd order features. As can be seen in table 2, The Decision Tree model has a low R Squared value of 0.191268249, indicating that it explains only 19.13% of the variance in the data. The RMSE value of 24.66102748 suggests that the model's predictions have a relatively high average error. The Random Forest model performs better than the Decision Tree model, with an improved R Squared value of 0.442148767. The RMSE value of 20.48179991 is lower than that of the Decision Tree model, indicating better prediction accuracy. The XGBoost model demonstrates the best performance among the three models, with a higher R Squared value of 0.658448994 and a lower RMSE value of 16.02642683. This indicates that the XGBoost model explains a larger portion of the variance in the data and provides more accurate predictions compared to the other two models.

Table 2. ML methods using raw features

Decision Tree (DT)	
R Squared	RMSE
0.191268	24.66103
Random Forest (RF)	
R Squared	RMSE
0.442149	20.4818
XGBoost	
R Squared	RMSE
0.658449	16.02643

Table 3. ML methods using 1st order features

Decision Tree (DT)	
R Squared	RMSE
0.448076	20.37271
Random Forest (RF)	
R Squared	RMSE
0.634719	16.57382
XGBoost	
R Squared	RMSE
0.857663	10.3459

Table 3 reports the results using first order features. The Decision Tree model shows an improved performance compared to the raw features. The R Squared value is now 0.448075651, indicating that the model explains approximately 44.81% of the variance in the data. The RMSE value of 20.37270503 suggests a lower average prediction error compared to the raw features model. The Random Forest model performs even better than the Decision Tree model for the 1st order features. The R Squared value increases to 0.634718882, indicating that the model explains approximately 63.47% of the variance in the data. The RMSE value decreases to 16.57381672, suggesting improved prediction accuracy compared to both the raw features and the Decision Tree model.

The XGBoost model shows the highest performance among the three models for the 1st order features. The R Squared value is significantly higher at 0.857662784, indicating that the model explains approximately 85.77% of the variance in the data. The RMSE value decreases to 10.34589707, indicating the highest level of prediction accuracy among the three models.

Table 3. ML methods using 2nd order features

Decision Tree (DT)	
R Squared	RMSE
0.235309	23.98015
Random Forest (RF)	
R Squared	RMSE
0.480075	19.77331
XGBoost	
R Squared	RMSE
0.634512	16.57851

The Decision Tree model shows a moderate improvement compared to the raw features. The R Squared value is now 0.23530877, indicating that the model explains approximately 23.53% of the variance in the data. The RMSE value of 23.98015439 suggests a lower average prediction error compared to the raw features model, but still relatively high. The Random Forest model performs better than the Decision Tree model for the 2nd order features. The R Squared value increases to 0.480074759, indicating that the model explains approximately 48.01% of the variance in the data. The RMSE value decreases to 19.77330983, suggesting improved prediction accuracy compared to both the raw features and the Decision Tree model.

The XGBoost model shows further improvement compared to both the Decision Tree and Random Forest models for the 2nd order features. The R Squared value increases to 0.634512005, indicating that the model explains approximately 63.45% of the variance in the data. The RMSE value decreases to 16.57850935, indicating improved prediction accuracy compared to both the raw features and the Decision Tree model.

Table 4. 1st and 2nd order features combined

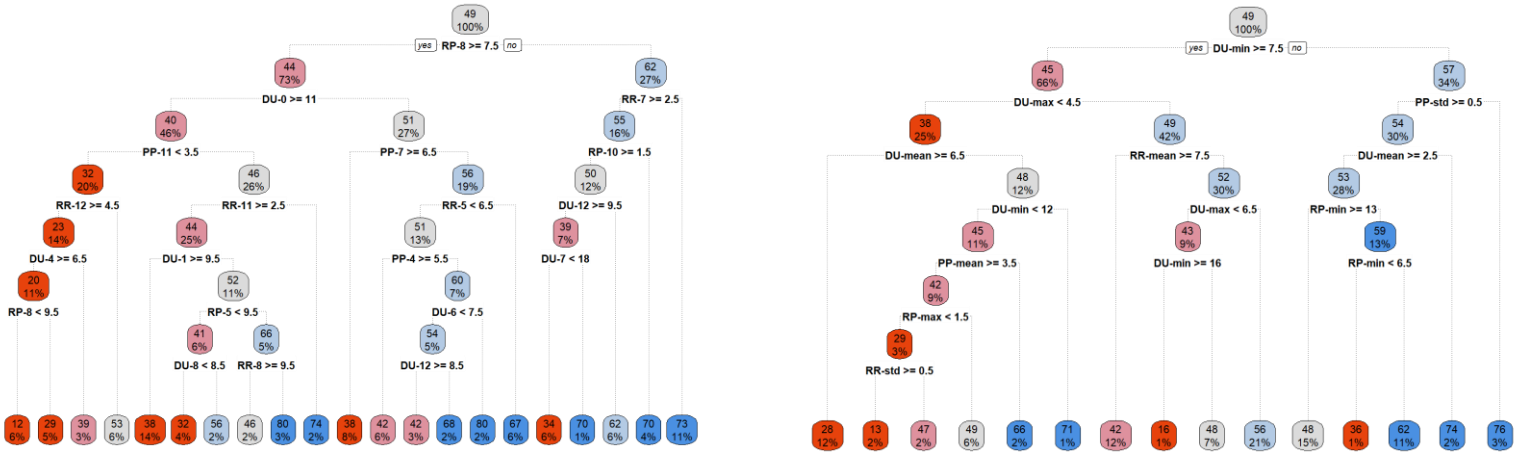
Decision Tree (DT)	
R Squared	RMSE
0.480283	19.76936
Random Forest (RF)	
R Squared	RMSE
0.629492	16.69198
XGBoost	
R Squared	RMSE
0.900191	8.663524

The Decision Tree model shows improved performance compared to both the raw features and the 2nd order features as can be seen table 4 and figure 3. The R Squared value is now 0.480282502, indicating that the model explains approximately 48.03% of the variance in the data. The RMSE value of 19.76935908 suggests a lower average prediction error compared to both the raw features and the 2nd order features. The Random Forest model

performs better than the Decision Tree model for the combined 1st and 2nd order features. The R Squared value increases to 0.629491582, indicating that the model explains approximately 62.95% of the variance in the data. The RMSE value decreases to 16.69198402, suggesting improved prediction accuracy compared to both the raw features, the 1st order features, and the 2nd order features.

The XGBoost model shows the highest performance among the three models for the combined 1st and 2nd order features. The R Squared value is significantly higher at 0.900190624, indicating that the model explains approximately 90.02% of the variance in the data. The RMSE value decreases to 8.663524149, indicating the highest level of prediction accuracy among the three models for this feature combination.

Figure 3. Decision trees for first order features (left) and second order features (right)



The raw features consist of recorded keystroke data, including the key-press time and key-release time for each key. These features provide the fundamental timing information of the key events. However, they might lack higher-level patterns and relationships that can be captured by more derived features. The first order features are derived from the raw data by calculating durations and latencies between key events. Durations represent the length of time a key is pressed, while latencies capture the time differences between key events. These features provide local characteristics of typing behavior and can capture individual key press patterns. The second order features are derived from the first order features and provide additional insights into typing behavior. Minimum and maximum values indicate the range of durations and latencies, while mean and standard deviation provide information about the average and variability of these timing measures. Slope features capture the overall pattern of typing, focusing on consistency in typing style. Spectral information, obtained through wavelet transformation, can reveal frequency components and further characterize the typing behavior.

The incorporation of first and second order features allows the models to capture more nuanced patterns and relationships in the keystroke data. While Decision Trees provide a baseline, Random Forest and XGBoost demonstrate superior performance due to their

ability to capture complex relationships. XGBoost, in particular, excels in leveraging the combined information from all the features, resulting in the highest level of explanatory power and prediction accuracy.

Conclusion

There is a growing need for robust and reliable user authentication methods in various domains, including computer systems, mobile devices, and online platforms. Traditional authentication methods such as passwords and PINs have limitations in terms of security and user convenience. Keystroke dynamics offer a potential solution by leveraging unique typing patterns inherent to individuals.

This research contributes to the field of keystroke dynamics by providing insights into the effectiveness of different feature extraction techniques and machine learning models for user authentication. The findings underscore the potential of keystroke dynamics as a reliable biometric authentication method and can inform the development of more robust and secure authentication systems in various domains.

One of the main limitations of this study is the size and composition of the dataset. Although it contains keystroke data from 50 users attempting to enter the string 'Restaurant' seven times, the relatively small number of users and repetitions may not fully capture the diversity and complexity of real-world typing behavior. A larger and more diverse dataset would provide a more comprehensive understanding of keystroke dynamics and potentially yield more accurate and reliable results.

The study was conducted under controlled conditions, where users were specifically instructed to enter the predefined string. This controlled environment may not fully represent real-world typing scenarios, where users interact with various applications and encounter different typing challenges. The study's findings might not generalize well to real-world settings, where typing patterns can be influenced by factors such as distractions, time pressure, or device variations. While these models demonstrated promising performance, their generalization to other datasets and real-world scenarios should be approached with caution. Different datasets with distinct characteristics and varying user populations might yield different model performance. Further evaluation and validation of the models on external datasets would strengthen the study's findings and establish the robustness of the proposed approach.

This study focused on analyzing keystroke dynamics for user authentication, assuming that individuals maintain consistent typing patterns over time. However, individual typing behavior can vary due to various factors such as physical and cognitive conditions, emotional states, or external influences. The study did not address the potential impact of user variability on authentication performance. Understanding the extent of user variability and its implications for keystroke dynamics-based authentication systems is an important aspect that should be considered in future research.

References

[1] R. Oak, “A Literature Survey on Authentication Using Behavioural Biometric Techniques,” in *Intelligent Computing and Information and Communication*, 2018, pp. 173–181.

[2] K. S. Killourhy and R. A. Maxion, “Comparing anomaly-detection algorithms for keystroke dynamics,” in *2009 IEEE/IFIP International Conference on Dependable Systems & Networks*, 2009, pp. 125–134.
[3] F. Monrose and A. D. Rubin, “Keystroke dynamics as a biometric for authentication,” *Future Gener. Comput. Syst.*, vol. 16, no. 4, pp. 351–359, Feb. 2000.
[4] C. Epp, M. Lippold, and R. L. Mandryk, “Identifying emotional states using keystroke dynamics,” in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, Vancouver, BC, Canada, 2011, pp. 715–724.
[5] F. Bergadano, D. Gunetti, and C. Picardi, “User authentication through keystroke dynamics,” *ACM Trans. Inf. Syst. Secur.*, vol. 5, no. 4, pp. 367–397, Nov. 2002.
[6] S. Bleha, C. Slivinsky, and B. Hussien, “Computer-access security systems using keystroke dynamics,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 12, no. 12, pp. 1217–1222, Dec. 1990.
[7] P. H. Pisani and A. C. Lorena, “A systematic review on keystroke dynamics,” *J. Braz. Comput. Soc.*, vol. 19, no. 4, pp. 573–587, Jul. 2013.
[8] J. Ilonen, “Keystroke dynamics,” *Advanced Topics in Information Processing–Lecture*, 2003.
[9] F. Monrose and A. Rubin, “Authentication via keystroke dynamics,” in *Proceedings of the 4th ACM conference on Computer and communications security*, Zurich, Switzerland, 1997, pp. 48–56.
[10] H. Crawford, “Keystroke dynamics: Characteristics and opportunities,” in *2010 Eighth International Conference on Privacy, Security and Trust*, 2010, pp. 205–212.
[11] P. S. Teh, A. B. J. Teoh, and S. Yue, “A survey of keystroke dynamics biometrics,” *ScientificWorldJournal*, vol. 2013, p. 408280, Nov. 2013.
[12] R. Giot, M. El-Abed, and C. Rosenberger, “Keystroke dynamics overview,” in *Biometrics*, J. Yang, Ed. London, England: InTech, 2011.
[13] P. Lison, “An introduction to machine learning,” *Language Technology Group (LTG)*, 2012.
[14] M. A. Veronin, R. Dixit, and R. P. Schumaker, “A Decision Tree Analysis of Opioid and Prescription Drug Interactions Leading to Death Using the FAERS Database,” in *IIMA/ICITED Joint Conference 2018*, 2018, pp. 67–67.
[15] O. Simeone, “A Brief Introduction to Machine Learning for Engineers,” *Found. Signal. Process. Commun. Netw.*, vol. 12, no. 3–4, pp. 200–431, 2018.
[16] G. Rebala, A. Ravi, and S. Churiwala, *An introduction to Machine Learning*. Berlin, Germany: Springer, 2019.
[17] Y. Başţanlar and M. Ozuysal, “Introduction to machine learning,” *Methods Mol. Biol.*, vol. 1107, pp. 105–128, 2014.
[18] R. R. Dixit, “Predicting Fetal Health using Cardiotocograms: A Machine Learning Approach,” *Journal of Advanced Analytics in Healthcare Management*, vol. 6, no. 1, pp. 43–57, 2022.
[19] S. Badillo *et al.*, “An Introduction to Machine Learning,” *Clin. Pharmacol. Ther.*, vol. 107, no. 4, pp. 871–885, Apr. 2020.
[20] O. Simeone, “A Very Brief Introduction to Machine Learning With Applications to Communication Systems,” *IEEE Transactions on Cognitive Communications and Networking*, vol. 4, no. 4, pp. 648–664, Dec. 2018.
[21] Y. Kodratoff, *Introduction to machine learning*. Elsevier, 2014.

[22] R. Y. Choi, A. S. Coyner, J. Kalpathy-Cramer, M. F. Chiang, and J. P. Campbell, "Introduction to Machine Learning, Neural Networks, and Deep Learning," *Transl. Vis. Sci. Technol.*, vol. 9, no. 2, p. 14, Feb. 2020.
[23] V. Kommaraju, K. Gunasekaran, K. Li, and T. Bansal, "Unsupervised pre-training for biomedical question answering," *arXiv preprint arXiv*, 2020.