Research Article

Analysis of Cloud Based Keystroke Dynamics for Behavioral Biometrics Using Multiclass Machine Learning

Manjunath Reddy
Customer Engineering Lead, Qualcomm , San diego ,CA, USA
reddymanjushari@gmail.com

Anusha Bodepudi
Principal Engineer, Workday, Frisco, Texas, USA
anusha.bodepudi@workday.com

Abstract

With the rapid proliferation of interconnected devices and the exponential growth of data stored in the cloud, the potential attack surface for cybercriminals expands significantly. Behavioral biometrics provide an additional layer of security by enabling continuous authentication and real-time monitoring. Its continuous and dynamic nature offers enhanced security, as it analyzes an individual's unique behavioral patterns in real-time. In this study, we utilized a dataset consisting of 90 users' attempts to type the 11-character string 'Exponential' eight times. Each attempt was recorded in the cloud with timestamps for key press and release events, aligned with the initial key press. The objective was to explore the potential of keystroke dynamics for user authentication. Various features were extracted from the dataset, categorized into tiers. Tier-0 features included key-press time and key-release time, while Tier-1 derived features encompassed durations, latencies, and digraphs. Additionally, Tier-2 statistical measures such as maximum, minimum, and mean values were calculated. The performance of three popular multiclass machine learning models, namely Decision Tree, Multi-layer Perceptron, and LightGBM, was evaluated

using these features. The results indicated that incorporating Tier-1 and Tier-2 features significantly improved the models' performance compared to relying solely on Tier-0 features. The inclusion of Tier-1 and Tier-2 features allows the models to capture more nuanced patterns and relationships in the keystroke data. While Decision Trees provide a baseline, Multi-layer Perceptron and LightGBM outperform them by effectively capturing complex relationships. Particularly, LightGBM excels in leveraging information from all features, resulting in the highest level of explanatory power and prediction accuracy. This highlights the importance of capturing both local and higher-level patterns in keystroke data to accurately authenticate users.

Keywords: *Keystroke dynamics, User authentication, Machine learning models, Tier-1 and Tier-2 features, LightGBM*

Introduction

The significance of automated biometric identification methods is progressively growing within both corporate and public security systems. These techniques play a vital role in ensuring the safety and integrity of various establishments. The term "biometric" originates from the Greek words "bio," which refers to life, and "metric," which means to measure [1]. In today's world, the reliance on biometric recognition systems has surged due to their ability to accurately identify individuals based on their unique physiological or behavioral characteristics. These systems leverage advanced technology to measure and analyze distinct features, such as fingerprints, facial patterns, iris structures, voice patterns, and even gait or typing patterns. By extracting and storing these distinctive attributes, biometric recognition systems enable swift and accurate identification of individuals, making them an invaluable tool for security purposes.

Behavioral biometrics involve the analysis of an individual's interactions with their devices, encompassing factors like their grip, typing speed, and other behavioral patterns. Through this analysis, a distinct behavioral profile is created, serving as a unique identifier for each person. Unlike active authentication methods that necessitate specific actions, such as facial recognition or entering a PIN, behavioral biometrics are considered passive since they can be seamlessly collected without requiring any additional user effort [2].

By observing how a person holds their devices, their typing rhythm, or even the pressure they apply while interacting with the device's touchscreen, behavioral biometrics capture an array of subtle behavioral cues that reflect an individual's habitual patterns. These patterns form a behavioral signature that distinguishes one user from another, allowing for reliable identification and authentication [3].

The advantage of behavioral biometrics lies in their unobtrusive nature. Users can be authenticated seamlessly in the background without the need for explicit actions

or prompts, enhancing user experience and convenience. Additionally, as these behavioral traits are unique to each individual, they provide an additional layer of security against unauthorized access.

Compared to traditional authentication methods like passwords or PINs, behavioral biometrics offer several benefits. They are difficult to replicate or guess since they rely on the individual's subconscious behaviors, making them more resistant to impersonation or hacking attempts. Furthermore, behavioral biometrics are less prone to being forgotten or misplaced, eliminating the need for frequent password resets or the risk of written credentials being stolen.

Behavioral biometric verification encompasses a range of methods that analyze various aspects of an individual's behavior to establish their unique identity[4], [5]. These methods include keystroke dynamics, gait analysis, voice identification, mouse usage characteristics, signature analysis, and cognitive biometrics [6]. By examining these behavioral traits, a person's distinct biometric profile can be established, enabling secure authentication. Keystroke dynamics involves analyzing an individual's typing patterns, such as the duration between keystrokes or the pressure applied while typing. Gait analysis focuses on the way a person walks, taking into account factors like stride length and cadence. Voice identification utilizes unique vocal characteristics, such as pitch and intonation, to verify a person's identity [7], [8]. Mouse usage characteristics analyze patterns in how a person moves and interacts with a computer mouse. Signature analysis examines the specific nuances and characteristics present in an individual's signature. Lastly, cognitive biometrics involve assessing cognitive processes like memory, attention, and decision-making patterns [9].

Keystroke dynamics is a biometric technique that utilizes the unique typing patterns, rhythm, and speed of an individual on a keyboard to create a biometric template for identification. This technique relies on two main measurements: dwell time and flight time [10], [11]. Dwell time refers to the duration for which a key is pressed down by the typist. It captures the time taken by an individual to press and hold a key before releasing it. On the other hand, flight time represents the duration between releasing one key and pressing the next key. It measures the time interval during which the typist's fingers are in the air, transitioning between keystrokes [3], [12].

By analyzing these raw measurements, keystroke dynamics can establish a distinct behavioral pattern specific to each individual. The time taken by a person to locate the correct key (flight time) and the duration they hold down a key (dwell time) exhibit individualistic characteristics that are independent of overall typing speed [13], [14]. Additionally, the rhythm with which certain sequences of characters are

typed can vary significantly from person to person. For instance, individuals accustomed to typing in English may exhibit faster typing speed for specific character sequences like "the," compared to someone with French language background [15], [16].

To enhance the accuracy and reliability of keystroke dynamics, some software combines this biometric technique with other user interactions on the computer. For example, it may consider mouse movements, such as acceleration time (how quickly the mouse pointer accelerates) and click frequency. By incorporating these additional interactions, the software can create a more comprehensive behavioral profile, improving the overall accuracy of identification and authentication processes [17]–[19]. The integration of keystroke dynamics with other computer interactions allows for a more holistic approach to behavioral biometrics [20], [21]. By analyzing multiple behavioral aspects, such as typing patterns, mouse movements, and clicking behavior, the software can build a more robust and accurate representation of an individual's unique behavioral traits, enhancing the effectiveness of the authentication process.

The versatility of behavioral biometrics allows them to be applied in various settings. Financial institutions, businesses, government facilities, and retail point of sale (POS) systems widely use behavioral biometrics for secure authentication purposes [22]. These environments prioritize the accurate identification of individuals to prevent unauthorized access, protect sensitive information, and combat identity theft or fraud. Furthermore, the adoption of behavioral biometrics is expanding to encompass a growing number of other sectors, as organizations recognize the value of leveraging behavioral characteristics to enhance security and streamline user authentication processes.

Feature engineering

Keystroke dynamics analysis involves extracting and examining different characteristics derived from recorded keystroke data.

Tier-0 features: The recorded keystroke data consists of key-press time and key-release time for each key [23].

Tier-1 features: The commonly extracted characteristics are known as 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 specific key "i," its duration is determined using the formula: duration = time when event = RELEASE - time when event = PRESS This calculation produces a timing vector, also referred to as PR, which contains the duration of each key press in the order they were pressed. PR(i) represents the duration of the "i-th" key press for all "i" where $1 \leq i \leq n$.

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

PP Latency: This measures the time difference between pressing each key. It is calculated using the equation: For all "i" where $1 \leq i < n$, PP(i) = time when (i+1)-th event = PRESS - time when i-th event = PRESS [23].

RR Latency: This represents the time difference between releasing each key. It is calculated using the equation: For all "i" where $1 \leq i < n$, RR(i) = time when (i+1)-th event = RELEASE - time when i-th event = RELEASE

RP Latency: This indicates the time difference between releasing one key and pressing the next key. It can be calculated using the equation: For all "i" where $1 \leq i < n$, RP(i) = time when (i+1)-th event = PRESS - time when i-th event = RELEASE

Another commonly encountered concept is the digraph, which represents the time required to press two keys sequentially. The digraph features (D) of a password are calculated as follows: For all "i" where $1 \leq i < n$, Di = time when (i+1)-th event = RELEASE - time when i-th event = PRESS.

Tier-2 features: Some characteristics are not directly derived from the raw biometric data but are instead derived from the first-order features [23].

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 typing pattern. We expect users to maintain a consistent typing style even if their speed varies. The new set of features is computed as follows: For all "i" where $1 \leq i < n$, result(i) = source(i+1) - source(i).

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

Table 1. Features

Tier	Features
Tier-0	Recorded keystroke data: key-press time and key-release time for each key
Tier-1	Local or first-order features:
	- Duration: length of time a key is pressed (calculated as RELEASE time - PRESS time)
	- PP Latency: time difference between pressing each key
	- RR Latency: time difference between releasing each key
	- RP Latency: time difference between releasing one key and pressing the next key
	- Digraph features (D): time required to press two keys sequentially
Tier-2	Derived features from Tier-1 features:
	- Minimum/maximum values: for latency and duration
	- Mean/standard deviation: for latency and duration
	- Slope: examining the overall pattern of typing
	- Spectral information: applying discrete wavelet transformation to extracted features

To preprocess our continuous data and convert it into discrete values, we utilized a technique called binning or discretization. Continuous data refers to variables that can take on any value within a certain range, such as age or income. However, many machine learning algorithms require discrete input, which means data points are divided into specific categories or bins. Binning allows us to transform the continuous data into a more manageable form without losing important information.

The binning process involves dividing the range of continuous feature values into distinct intervals or bins and assigning discrete labels to each bin. This categorization simplifies the representation of our data and makes it more suitable for certain algorithms. For example, if we have a dataset with ages ranging from 20 to 60, we can create bins like [20-30), [30-40), [40-50), [50-60), where the brackets indicate inclusive and exclusive boundaries. Each individual's age would then be assigned to one of these bins, effectively converting the continuous age values into discrete categories.

To perform the binning process, we employed a specific algorithm called KBinsDiscretizer. This algorithm is designed to handle the discretization task effectively and efficiently. It takes the continuous features as input and outputs

discrete values that fall within the desired range of [0, numBins). The numBins parameter represents the number of bins or intervals we want to create. By specifying the desired number of bins, we can control the granularity of the discretization process. For example, if we set numBins to 5, the algorithm will create five equally sized bins, dividing the range of continuous values into five intervals.

After discretizing our features using KBinsDiscretizer, the next step was to split our data into two sets: the training dataset and the validation dataset [24]. This division plays a critical role in evaluating the performance of classifiers or predictive models. To ensure that the data distribution is maintained during the split, we employed stratified sampling. This sampling technique is designed to create subsets that accurately represent the class distribution 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. Stratified sampling helps prevent any significant imbalances in class representation between the training and validation datasets, ensuring that our evaluation is reliable and unbiased.

Classification algorithms

The Decision Tree algorithm is a popular and widely used machine learning algorithm that is used for both classification and regression tasks [25]. It is a supervised learning algorithm that builds a model in the form of a tree structure, where each internal node represents a feature or attribute, each branch represents a decision rule, and each leaf node represents the outcome or prediction. The algorithm learns from a training dataset by recursively partitioning the data based on the selected features, aiming to maximize the information gain or minimize impurity at each step. Decision trees are easily interpretable and can handle both categorical and numerical data, making them suitable for a wide range of applications [26].

The Multi-layer Perceptron (MLP) algorithm is a type of artificial neural network that is widely used for both classification and regression tasks [27]. It consists of multiple layers of interconnected nodes, or artificial neurons, organized into an input layer, one or more hidden layers, and an output layer. Each neuron in the network applies a nonlinear activation function to the weighted sum of its inputs, allowing the model to capture complex relationships between the features and the target variable [28]. The weights and biases of the neurons are learned through a process called backpropagation, which iteratively adjusts the parameters to minimize the difference between the predicted and actual outputs.

MLPs are known for their ability to learn complex and nonlinear patterns in the data, making them particularly useful in tasks where the relationship between the

input and output variables is not easily captured by linear models [28]. They can handle both continuous and categorical features, as well as handle missing data by utilizing techniques such as imputation. MLPs are also capable of automatically extracting relevant features from raw data through the hidden layers, allowing them to discover high-level representations that are useful for prediction. Additionally, MLPs can be trained on large-scale datasets using techniques like mini-batch gradient descent and parallel computing [29].

LightGBM is a gradient boosting framework that is designed to provide fast and efficient training of gradient boosting models. It is an open-source library developed by Microsoft, specifically optimized for large-scale datasets and high-dimensional feature spaces [30], [31]. LightGBM utilizes the gradient boosting algorithm, which combines multiple weak predictive models (typically decision trees) into a strong ensemble model. However, LightGBM introduces several optimizations to enhance training speed and memory efficiency. It uses a technique called histogram-based gradient boosting, where the data is binned into histograms to reduce the memory consumption and speed up the training process.

Results and discussion

We calculated R-square and RMSE to evaluate The Decision Tree, Multi-layer Perceptron, and LightGBM. 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.

Table 2. Tier-0 features

Algorithm	R Squared	RMSE
Decision Tree	0.202	25.772
Multi-layer Perceptron	0.553	21.493
LightGBM	0.769	17.538

In Table 2, the tier-0 features of different algorithms are presented along with their corresponding R-squared and root mean square error (RMSE) values. The Decision Tree algorithm achieved an R-squared value of 0.202, indicating that it explains only a small portion of the variance in the data. The RMSE value of 25.772 suggests that the predictions of this algorithm have a relatively high average difference from the actual values. The Multi-layer Perceptron algorithm performed better, with an R-squared value of 0.553. This indicates a moderate level of fit to the data. The RMSE value of 21.493 suggests that the predictions of this algorithm have a lower average difference from the actual values compared to the Decision

Tree algorithm. The LightGBM algorithm achieved the highest R-squared value of 0.769, indicating a relatively strong fit to the data. Additionally, the RMSE value of 17.538 suggests that the predictions of this algorithm have a lower average difference from the actual values compared to both the Decision Tree and Multi-layer Perceptron algorithms.

Figure 1. Decision tree with Tier-2 features.

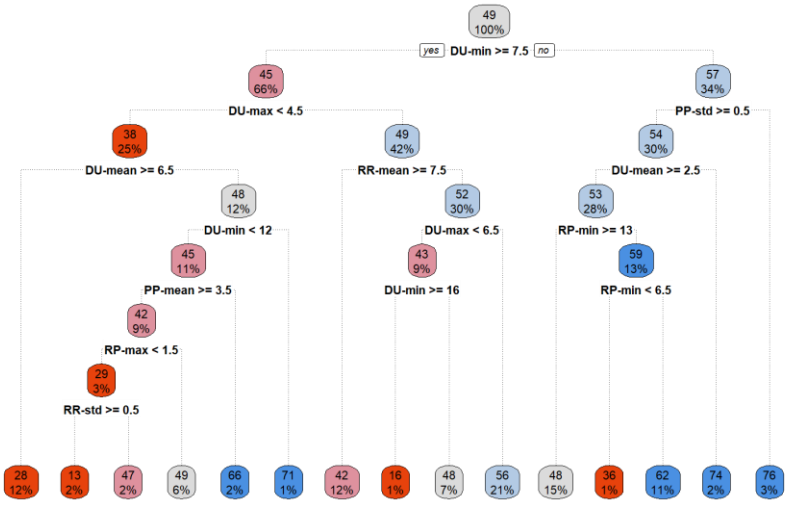


Table 3. Tier-1 features

Algorithm	R Squared	RMSE
Decision Tree	0.559	21.484
Multi-layer Perceptron	0.746	17.685
LightGBM	0.969	11.346

Table 3 presents the tier-1 features of different algorithms, along with their corresponding R-squared and root mean square error (RMSE) values. The Decision Tree algorithm achieved an R-squared value of 0.559, indicating that it explains a moderate portion of the variance in the data. The RMSE value of 21.484 suggests that the predictions of this algorithm have a relatively low average difference from the actual values. The Multi-layer Perceptron algorithm performed

better, with an R-squared value of 0.746. This indicates a relatively strong fit to the data. The RMSE value of 17.685 suggests that the predictions of this algorithm have a lower average difference from the actual values compared to the Decision Tree algorithm. The LightGBM algorithm achieved the highest R-squared value of 0.969, indicating an excellent fit to the data.

Table 4. Tier-2 features:

Algorithm	R Squared	RMSE
Decision Tree	0.346	24.891
Multi-layer Perceptron	0.591	20.764
LightGBM	0.746	17.678

In Table 4, the Tier-2 features are presented along with their corresponding evaluation metrics. Decision Tree, achieved an R Squared value of 0.346, indicating that it explains approximately 34.6% of the variance in the data. The Root Mean Squared Error (RMSE) for this algorithm is 24.891. LightGBM, demonstrates the highest performance among the three, with an R Squared value of 0.746. It can be inferred that LightGBM outperforms both Decision Tree and Multi-layer Perceptron in terms of both R Squared and RMSE, making it the most effective algorithm for the given task.

Table 5. Tier-1 and tier-2 combined:

Algorithm	R Squared	RMSE
Decision Tree	0.591	20.765
Multi-layer Perceptron	0.731	17.792
LightGBM	0.913191	9.664

The results in table 5 show that the Decision Tree algorithm achieved an R Squared value of 0.591 and an RMSE value of 20.765. The Multi-layer Perceptron algorithm performed better with an R Squared value of 0.731 and an RMSE value of 17.792. However, the LightGBM algorithm outperformed both with an impressive R Squared value of 0.913190521 and an RMSE value of 9.664. These results suggest that the LightGBM algorithm exhibited the highest level of accuracy and predictive power among the three algorithms evaluated in both Tier-1 and Tier-2. the LightGBM algorithm consistently outperformed both the

Decision Tree and Multi-layer Perceptron algorithms in all tiers, indicating its superiority in handling different feature sets.

The Tier-0 features in keystroke dynamics analysis encompass the basic recorded keystroke data, specifically the timing information of key-press and key-release events for each key. These features serve as the foundation, providing essential temporal data. However, they may have limitations in capturing more complex patterns and relationships that exist in the typing behavior.

To address this, Tier-1 features are derived from the raw data by analyzing durations and latencies between key events. Durations refer to the duration of time a key remains pressed, while latencies represent the time differences between different key events. By calculating these measures, Tier-1 features offer insights into local characteristics of typing behavior, allowing for the identification of individual key press patterns.

Tier-2 features are derived from the Tier-1 features, providing additional layers of information and a deeper understanding of typing behavior. These features include metrics such as minimum and maximum values, which indicate the range of durations and latencies observed. Mean and standard deviation are calculated to determine the average and variability of these timing measures, offering insights into the central tendency and spread of the data.

Another aspect considered in Tier-2 features is the slope. By analyzing the slope of the biometric sample, it is possible to focus on the overall typing pattern rather than individual key events. This helps in assessing the consistency of typing style, allowing for the detection of characteristic typing behaviors even if the typing speed varies. Furthermore, spectral information is obtained by applying wavelet transformation to the previously extracted features. This process reveals frequency components and enables the characterization of typing behavior in terms of different frequency ranges.

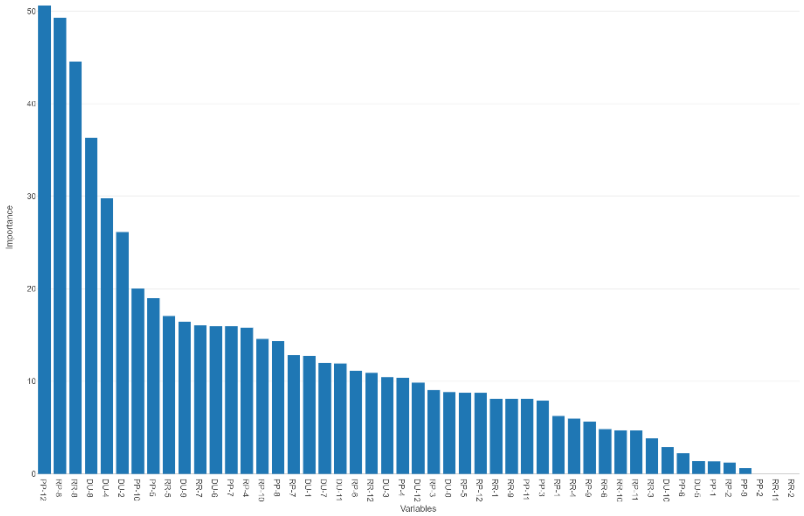
The inclusion of first and Tier-2 features in keystroke dynamics analysis significantly enhances the models' ability to discern intricate patterns and relationships within the recorded keystroke data. While Tier-0 features offer fundamental timing information, they may not fully capture the higher-level complexities present in typing behavior. By incorporating first-order features, such as durations and latencies, the models gain access to local characteristics that provide a more granular understanding of individual key press patterns.

In evaluating the performance of different algorithms, Decision Trees serve as a baseline for comparison. However, the Multi-layer Perceptron and LightGBM algorithms exhibit superior performance. This can be attributed to their capacity to

capture and leverage the complex relationships between the derived features. The Multi-layer Perceptron algorithm, with its layered structure and nonlinear activation functions, demonstrates a better fit to the data compared to the Decision Tree algorithm. It effectively learns and represents the intricate patterns present in the keystroke data, resulting in higher explanatory power and more accurate predictions.

Among the algorithms considered, LightGBM stands out as the top performer. Its ability to effectively leverage all the available features, including both first and Tier-2 features, contributes to its exceptional performance. LightGBM excels in capturing and utilizing the combined information from these features, allowing it to uncover complex patterns and relationships that may not be apparent when considering the features individually. This comprehensive approach results in the highest level of explanatory power and prediction accuracy among the evaluated algorithms, making LightGBM an ideal choice for keystroke dynamics analysis tasks.

Figure 1. Feature importance



Conclusion

The research on utilizing behavioral biometrics, specifically keystroke dynamics, for user authentication holds significant importance in the field of cybersecurity and authentication methods. With the increasing prevalence of AI, IoT, and cloud computing, traditional authentication methods such as passwords and PINs are

becoming less reliable. Behavioral biometrics offers a novel approach by analyzing an individual's unique behavioral patterns in real-time, providing enhanced security.

In this study, we focused on keystroke dynamics and collected a dataset consisting of 90 users' attempts to type a specific string multiple time. By recording key press and release events along with timestamps, we extracted various features categorized into tiers. Tier-0 features included basic key-press and key-release times, while Tier-1 features encompassed durations, latencies, and digraphs. Tier-2 features involved statistical measures like maximum, minimum, and mean values.

The research findings emphasized the importance of incorporating Tier-1 and Tier-2 features in the authentication models. Compared to relying solely on Tier-0 features, the inclusion of more nuanced patterns and relationships in the keystroke data significantly improved the models' performance. This highlights the need to capture both local and higher-level patterns to accurately authenticate users.

One significant contribution of this study is the identification and evaluation of the importance of different tiers of features in keystroke dynamics for user authentication. By categorizing the features into Tier-0, Tier-1, and Tier-2, were able to systematically analyze their impact on the performance of authentication models. This contribution sheds light on the fact that relying solely on basic key-press and key-release times (Tier-0 features) may not be sufficient to accurately authenticate users. The inclusion of more advanced features, such as durations, latencies, digraphs, and statistical measures (Tier-1 and Tier-2 features), significantly improved the models' performance. This finding underscores the importance of capturing and utilizing more nuanced patterns and relationships in keystroke data to enhance the accuracy and reliability of user authentication systems.

The dataset used in this research consisted of a relatively small sample size of 90 users. While efforts were made to collect data from a diverse group, the generalizability of the findings may be limited. A larger and more diverse dataset would provide a more comprehensive understanding of the performance of the authentication models and ensure the reliability of the results across different user populations.

References

[1] 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.

[2] R. R. O. Al-Nima, S. S. Dlay, and W. L. Woo, “A new approach to predicting physical biometrics from behavioural biometrics,” *International Journal of Computer and Information Engineering*, vol. 8, no. 11, pp. 2001–2006, 2014.

[3] R. Giot, M. El-Abed, and C. Rosenberger, “Keystroke dynamics overview,” in *Biometrics*, J. Yang, Ed. London, England: InTech, 2011.

[4] F. Cherifi, B. Hemery, R. Giot, M. Pasquet, and C. Rosenberger, “Performance Evaluation of Behavioral Biometric Systems,” in *Behavioral Biometrics for Human Identification: Intelligent Applications*, IGI Global, 2010, pp. 57–74.

[5] M. Wolff, “Behavioral Biometric Identification on Mobile Devices,” in *Foundations of Augmented Cognition*, 2013, pp. 783–791.

[6] R. Oak and M. Khare, “A Novel Architecture for Continuous Authentication using Behavioural Biometrics,” in *2017 International Conference on Current Trends in Computer, Electrical, Electronics and Communication (CTCEEC)*, 2017, pp. 767–771.

[7] R. V. Yampolskiy and V. Govindaraju, “Taxonomy of Behavioural Biometrics,” in *Behavioral Biometrics for Human Identification: Intelligent Applications*, IGI Global, 2010, pp. 1–43.

[8] H. Saevanee and P. Bhatarakosol, “User Authentication Using Combination of Behavioral Biometrics over the Touchpad Acting Like Touch Screen of Mobile Device,” in *2008 International Conference on Computer and Electrical Engineering*, 2008, pp. 82–86.

[9] P. S. Teh, A. B. J. Teoh, and S. Yue, “A survey of keystroke dynamics biometrics,” *ScientificWorldJournal*, vol. 2013, p. 408280, Nov. 2013.

[10] 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.

[11] A. Buriro, B. Crispo, and M. Conti, “AnswerAuth: A bimodal behavioral biometric-based user authentication scheme for smartphones,” *Journal of information security and*, 2019.

[12] 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.

[13] P. M. Koh and W. K. Lai, “Keystroke Dynamics Identification System using ABC Algorithm,” in *2019 IEEE International Conference on Automatic Control and Intelligent Systems (I2CACIS)*, 2019, pp. 108–113.

[14] S. Maheshwary and V. Pudi, “Mining Keystroke Timing Pattern for User Authentication,” in *New Frontiers in Mining Complex Patterns*, 2017, pp. 213–227.

[15] P. S. Teh, A. B. J. Teoh, T. S. Ong, and H. F. Neo, "Statistical Fusion Approach on Keystroke Dynamics," in *2007 Third International IEEE Conference on Signal-Image Technologies and Internet-Based System*, 2007, pp. 918–923.
[16] J. Ho and D.-K. Kang, "One-class naïve Bayes with duration feature ranking for accurate user authentication using keystroke dynamics," *Applied Intelligence*, vol. 48, no. 6, pp. 1547–1564, Jun. 2018.
[17] J. Ilonen, "Keystroke dynamics," *Advanced Topics in Information Processing–Lecture*, 2003.
[18] 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.
[19] 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.
[20] 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.
[21] 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.
[22] R. V. Yampolskiy and V. Govindaraju, "Behavioural biometrics: a survey and classification," *Int. J. Biom.*, vol. 1, no. 1, pp. 81–113, Jan. 2008.
[23] K. Daimi, G. Francia III, and L. H. Encinas, *Breakthroughs in digital biometrics and forensics*. Cham, Switzerland: Springer Nature, 2022.
[24] D. Polzer, "7 of the Most Used Feature Engineering Techniques Hands-on Feature Engineering with Scikit-Learn, Tensorflow, Pandas and Scipy," *towardsdatascience.com*.
[25] 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.
[26] Y. Kodratoff, *Introduction to machine learning*. Elsevier, 2014.
[27] E.-C. Kim, E.-Y. Kim, H.-C. Lee, and B.-J. Yoo, "The Details and Outlook of Three Data Acts Amendment in South Korea: With a Focus on the Changes of Domestic Financial and Data Industry," *Informatization Policy*, vol. 28, no. 3, pp. 49–72, 2021.
[28] 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.
[29] V. Kommaraju, K. Gunasekaran, K. Li, and T. Bansal, "Unsupervised pre-training for biomedical question answering," *arXiv preprint arXiv*, 2020.
[30] P. Lison, "An introduction to machine learning," *Language Technology Group (LTG)*, 2012.

[31] O. Simeone, “A Brief Introduction to Machine Learning for Engineers,” *Found. Signal. Process. Commun. Netw.*, vol. 12, no. 3–4, pp. 200–431, 2018.