

# Evaluating Keystroke Dynamics Performance in e-Commerce

Xiaofei Wang<sup>1</sup><sup>a</sup>, Andy Meneely<sup>2</sup><sup>b</sup> and Daqing Hou<sup>2</sup><sup>c</sup>

<sup>1</sup>Department of Electrical & Computer Engineering, Clarkson University, Potsdam, NY, U.S.A.

<sup>2</sup>Department of Software Engineering, Rochester Institute of Technology, Rochester, NY, U.S.A.  
wangx4@clarkson.edu, {andy.meneely, daqing.hou}@rit.edu

**Keywords:** Keystroke Dynamics, Dataset, Behavioral Biometrics, Optimization Algorithm, e-Commerce.

**Abstract:** The traditional username and password authentication mechanisms are vulnerable to various attacks, such as brute force, rainbow tables, and password theft. Multi-factor authentication is becoming the standard practice across the software industry, and keystroke dynamics can be a useful way to augment existing authentication mechanisms. This paper introduces a keystroke dynamics-based system implemented using the Django framework to collect and analyze keystroke data across three e-Commerce web services: air ticketing, online shopping, and car rental systems. Our system asked users to type their own information and also type several other users' information, using common and service-specific input fields. We collected data from 62 participants where each contributes 10 records for each service as both genuine and imposter users. Through detailed feature extraction and machine learning-based analysis with three binary classifiers, we evaluate the efficacy of keystroke dynamics in distinguishing genuine from imposter users. Our results indicate that different input fields have differentiated effects on verifying users, and appropriate field selection strategies can improve the performance of classification methods.


## 1 INTRODUCTION


The security of online systems has become a critical concern in today's digital landscape. Traditional authentication mechanisms rely heavily on knowledge-based factors such as usernames and passwords. However, these methods are increasingly vulnerable to a wide range of attacks, including phishing, brute force, theft, and rainbow tables. The widespread use of weak, reused, or easily guessable passwords continues to undermine user authentication. To mitigate the risks associated with passwords, web services have introduced secondary authentication mechanisms, such as security questions or one-time passcodes (OTPs). While these methods add an extra layer of security, they too have limitations, as they can be bypassed through social engineering or other forms of attack. Furthermore, these methods degrade usability by requiring additional steps in the authentication process, creating friction for the end user.


*Keystroke dynamics*, a form of behavioral biometrics, has emerged as a promising methodology for enhancing user authentication. Unlike traditional

knowledge-based or token-based methods, keystroke dynamics relies on each user's unique typing patterns. This modality leverages each individual's typing features such as the time interval between consecutive key presses, key hold times, and other timing-based characteristics. Since the typing behavior tends to be distinct, keystroke dynamics offers the potential for continuous authentication that is more resistant to conventional attacks. Keystroke dynamics is also non-intrusive, as it can be captured passively in the background without requiring additional hardware or user input (Wahab et al., 2023).

*The goal of this work is to improve the user authentication experience by collecting keystroke dynamics data in realistic scenarios.* We chose e-Commerce as a common scenario that many people regularly engage with, such as typing sensitive information, such as usernames, passwords, phone numbers, and credit card details. We developed a keystroke dynamics-based system in Python's Django framework. We designed three web services: an air ticket service system, an online shopping system, and a car rental service system. Each of these systems contains common input fields, such as *username*, *password*, and *phone number*, as well as service-specific fields, such as *car license plate* and *credit card number*. We

<sup>a</sup> <https://orcid.org/0009-0008-4412-4917>

<sup>b</sup> <https://orcid.org/0000-0002-4850-1408>

<sup>c</sup> <https://orcid.org/0000-0001-8401-7157>

investigated a dual input scenario where users provide both their own genuine information and imposter information, enabling a comparison of keystroke patterns between authentic and false inputs.

The key innovation of this system lies in its ability to capture and analyze keystroke data across multiple services and multiple input types, allowing us to explore how well keystroke dynamics can differentiate between *genuine inputs*, where users input their own data, and *imposter inputs*, where users input the data of another person. Our contributions are:

- a *keystroke data collection system* that captures keystroke dynamics across three different web services, each with common and service-specific input fields.
- a comprehensive *field-level analysis* to assess the effectiveness of keystroke dynamics in distinguishing authentic from imposter inputs, identifying which input fields (e.g., usernames, email addresses, credit card numbers) are most effective for this purpose.

## 2 RELATED WORK

Table 1: Public keystroke Datasets.

Dataset	Users	Pub. Year	Free Text?
(Dowland and Furnell, 2004)	35	2004	Yes
(Gunetti and Picardi, 2005)	205	2005	Yes
(Killourhy and Maxion, 2009)	51	2009	No
(Messerman et al., 2011)	55	2011	Yes
(Monaco et al., 2012)	30	2012	No
(Ahmed and Traore, 2013)	53	2013	Yes
(Vural et al., 2014)	39	2014	No
(Sun et al., 2016)	148	2016	No
Our Dataset	62	2024	Yes

In the field of keystroke dynamics research, several datasets have provided valuable insights into understanding user typing behavior.

The Torino dataset (Gunetti and Picardi, 2005) was collected in Italian. Participants were required to open an HTML form in a browser and freely input content they felt comfortable with. Each session generated approximately 800 keystrokes, totaling 400,000 keystrokes recorded from 40 participants. Additionally, 165 participants contributed data from

only one session, primarily for simulating imposter attacks, which provides a significant perspective on the security aspects of keystroke dynamics. Although this dataset captures individual typing habits in a natural environment, the limitation of recording only key press times without release times restricts the calculation of dwell times for individual keys, thereby hindering a deeper analysis of keystroke features.

In contrast, the Clarkson dataset was collected under laboratory conditions, involving 39 participants (Vural et al., 2014). Each participant completed two sessions, tasked with answering survey questions designed around their areas of interest to ensure fluidity and naturalness in their responses. This dataset recorded a total of 840,000 keystrokes, providing a rich array of free text samples. However, while the diversity and richness of the data are high due to the controlled environment, the limitations of the laboratory setting may compromise the authenticity of participant performance, thus affecting the external validity of the data.

The Buffalo dataset includes keystroke data from 148 participants, characterized by a mix of fixed text and free text input (Sun et al., 2016). Participants typed in a laboratory setting using four different types of keyboards, generating a total of 2.14 million keystrokes. This dataset explores the impact of keyboard type on keystroke dynamics, adding complexity to the experiments. However, the fixed text component may, to some extent, limit the breadth and depth of free text analysis.

The study (Killourhy and Maxion, 2009) focused on fixed password entry, utilizing a database containing 20,400 samples collected from 51 participants inputting the same fixed password, “.tie5Roanl.” Each participant contributed 400 samples. While the study reported that the Scaled Manhattan, Nearest Neighbor (Mahalanobis), and Outlier Count algorithms performed best, with an error rate (EER) ranging from 9.6% to 10.2%, the use of a fixed password may not accurately reflect user behavior when entering their actual passwords.

In comparison to the datasets in Table 1, the Django dataset employed in this study was collected through online deployment, allowing participants to access the collection interface via a URL link. This approach enabled participants to input data in their most familiar environment, enhancing the authenticity of the data. Furthermore, participants could freely decide their own input as well as mimic others’ inputs, which is crucial for distinguishing between genuine and imposter inputs. Once recorded, the data was sent back to a server for centralized storage. This dataset includes 62 participants and a total of 1.3M

keystrokes. Such a collection method ensures data authenticity and diversity, more accurately reflecting users' true typing habits. In summary, while existing datasets have provided important insights into keystroke dynamics, ours, by simulating real-life scenarios, advances a deeper analysis of keystroke dynamics in the common e-Commerce scenarios.

### 3 SYSTEM DESIGN

#### 3.1 System Architecture

The system consists of three primary components: the front-end for keystroke data collection, the Django back-end for processing and storing the data, and the database for managing user information and keystroke records. The architecture is designed to handle both desktop and mobile environments, though the primary focus of this work is on desktop interactions. The three simulated web services used for data collection include:

1. Air Ticket Service System: A web interface where users enter personal information (e.g., name, email) to book airline tickets.
2. Online Shopping System: A shopping cart system where users provide their shipping address, payment information, and contact details to complete a purchase.
3. Car Rental Service System: A rental booking system where users input their driver's license number, car preferences, and payment details.

Each service contains a mix of shared input fields and service-specific input fields. This combination allows for a rich dataset that covers both common and context-specific inputs.

#### 3.2 Keystroke Data Collection

The collection of keystroke dynamics is achieved using JavaScript embedded within the HTML forms of each web service. As users interact with the input fields, JavaScript captures the Keydown events, when a key is pressed down; and Keyup, when a key is released. For each key event, a timestamp is recorded, which allows for the extraction of various keystroke dynamics features. The data captured during these events is transmitted asynchronously (via AJAX) to the Django back-end, where it is processed and stored for further analysis.

To ensure robust data collection, users are asked to interact with the three web services multiple times,

Table 2: Input Fields in the three web services.

Field	Air Ticket Service	Online Shopping	Car Rental Service
Name	✓	✓	✓
Email	✓	✓	✓
Confirm Email	✓	✓	✓
Phone Number	✓	✓	✓
Gender	✓	×	×
Birthdate	✓	×	×
Card Holder	✓	✓	×
Card Number	✓	✓	×
Card Expiration	✓	✓	×
Security Code	✓	✓	×
Country	✓	✓	✓
Address	✓	✓	✓
City	✓	✓	✓
State	✓	✓	✓
ZIP	✓	✓	✓
Driver License	×	×	✓
License Expiration	×	×	✓
Issuing Authority	×	×	✓
Password	✓	✓	✓

both entering their own information (authentic input) and impersonating other users by entering data retrieved from the database (imposter input). This simulates a real-world scenario where an attacker might attempt to impersonate a legitimate user by entering known credentials, but with subtle differences in typing behavior.

#### 3.3 Keystroke Features

Once the raw keystroke data is captured, several key timing features are extracted to characterize the user's typing behavior:

- H (Dwell Time): The time a key is held down, measured from the moment the key is pressed (keydown event) to the moment the same key is released (keyup event). This is a key feature in keystroke dynamics as different users tend to have unique key hold times.
- PP (Key Press Interval): The time between two successive key presses, which captures the typing rhythm, and can vary significantly between users.
- PR (Flight Time): The time between releasing one key and pressing the next. This interval is often influenced by the cognitive and physical characteristics of the typist, providing a distinguishing feature.

These keystroke features are extracted for each individual input field across all web services, allowing us to analyze the typing patterns associated with different types of information (e.g., username, password, phone number). The extracted features are stored in CSV format, with each CSV file corresponding to a specific input field. This granular approach enables a detailed comparison of keystroke dynamics across different types of input.

### 3.4 Backend Data Processing

Once the raw keystroke data is transmitted from the front-end, the Django back-end processes it for storage and further analysis. The key steps in this process are as follows:

1. **Data Reception:** The keystroke data, including the timestamps of each keydown and keyup event, is sent via AJAX to Django's view functions, where it is immediately recorded.

2. **Feature Extraction:** The raw timestamps are processed to compute the dwell time (H), key press interval (PP), and flight time (PR) for each key event. The extracted features are then organized by input field and stored in separate CSV files for each user session.

3. **Data Storage:** The CSV files are saved in a structured format within the Django framework, with metadata including the user ID, session type (authentic or imposter), and timestamp. This structure allows for easy retrieval and analysis of the data during the model training phase.

### 3.5 Handling Imposter Data

A unique aspect of the system design is the inclusion of imposter scenarios, where users are asked to input data belonging to another user. This creates two distinct input data for each user: 1. Authentic Input Data where users input their own personal information. and 2. Imposter Input Data where users input information from another user.

The system tracks these two types of inputs and stores the corresponding keystroke dynamics for later comparison. By analyzing the differences between the authentic and imposter input patterns, the system aims to uncover which fields (e.g., phone number, email, credit card) exhibit the most significant variations, helping to identify potential attack vectors where imposter might be detectable.

### 3.6 System Workflow

The system operates as follows:

1. **User Interaction:** The user interacts with the three web services, either entering their own information or impersonating another user.
2. **Keystroke Data Collection:** JavaScript captures the keystroke events (keydown and keyup) and sends them to the Django server.
3. **Feature Extraction:** The Django server computes key features such as H (dwell time), PP (press-interval), and PR (press-release interval).
4. **Data Storage:** The extracted features are saved in CSV files for each input field and session, tagged with metadata indicating whether the input was authentic or imposter.
5. **Model Training and Prediction:** In the next stage (not covered in this section), machine learning models are trained on the collected data to distinguish between genuine and imposter inputs based on the keystroke dynamics.

This system design facilitates the collection of detailed keystroke data across multiple input fields and services, enabling a comprehensive analysis of user behavior in both normal and imposter scenarios.

## 4 EXPERIMENTAL SETUP AND RESULTS ANALYSIS

In this section, we describe the experimental setup used for collecting and analyzing the keystroke dynamics data, as well as the results from three experiments designed to evaluate the effectiveness of different features and feature combinations for distinguishing between genuine and imposter inputs. The experiments were conducted on a dataset that was specifically designed for this study, and the results are analyzed to determine which field contribute most significantly to the classification accuracy.

### 4.1 Dataset Overview

Each user generated 60 entries for each input field, of which 30 were authentic inputs (the user's own information), and the other 30 entries were imposter inputs (keystrokes from five imposter users, 6 entries per imposter user). A total of 62 users completed data collection, resulting in a dataset with a total of 1.35M keystrokes.

Most of the participants in this data collection are concentrated in the 18-40 age group (61), with only one in the 41-59 age group, and none the 60 and above group. In terms of gender distribution, male participants accounted for a large proportion (44), with rel-

atively fewer females (18). Most of the participants consider their typing skills intermediate (29) or advanced (28), with only 5 beginners. This result is consistent with the general trend of having younger participants with higher typing skills.

The dataset used for this study was collected from users interacting with three web services: an air ticket service system, an online shopping system, and a car rental service system. Each user provided input for common fields (e.g., username, password, phone number) as well as service-specific fields. Keystroke dynamics data, including Dwell Time (H), Key Press Interval (PP), and Flight Time (PR), were captured for each field using JavaScript.

## 4.2 Classifiers Used in the Experiments

In this study, we used three different classifiers to evaluate the effectiveness of various keystroke dynamics characteristics to distinguish between genuine and imposter inputs. The first classifier, the Decision Tree (DT), builds a predictive model by recursively splitting the dataset based on feature values, ultimately creating a structure that can be used to classify new input instances. The second classifier, the Support Vector Machine (SVM), works by identifying an optimal hyperplane that separates the data points into distinct classes, maximizing the margin between genuine and imposter inputs. Lastly, the Multilayer Perceptron (MLP), a type of artificial neural network, leverages multiple layers of interconnected neurons to learn complex patterns in the data through iterative training, making it well-suited for capturing nonlinear relationships in keystroke dynamics. These three classifiers were chosen to provide a comprehensive evaluation of both simple and complex decision-making mechanisms in classifying user input behavior. Furthermore, all classifiers were optimized using Grid Search to determine the best hyperparameters, ensuring optimal performance.

## 4.3 Experimental Design and Results

To evaluate the effectiveness of keystroke dynamics for detecting imposter, three experiments were conducted. These experiments aimed to assess the predictive power of individual keystroke features, identify the impact of removing specific features, and optimize feature combinations using a genetic algorithm.

### 4.3.1 Field-Level Analysis

In the first experiment, we aimed to evaluate the effectiveness of individual field keystroke features in

Table 3: Accuracy of Classifiers on Input Fields.

Feature	DT (%)	SVM (%)	MLP (%)
Name	87.63	92.07	93.01
Email	91.37	96.24	95.56
Phone Number	84.68	89.25	90.19
Country	85.35	82.39	86.56
Address	90.32	91.67	93.41
City	89.65	90.19	92.88
State	79.57	70.43	81.18
ZIP	85.75	85.62	91.52
Password	83.05	86.01	88.70

Table 4: Accuracy Change (%) After Removing One Field.

Removed field	DT	SVM	MLP
Name	-1.48	-0.27	0.69
Email	1.18	-0.41	-0.27
Phone Number	0.27	-1.32	0.00
Country	0.00	0.14	0.14
Address	1.48	0.41	0.27
City	0.59	-0.27	-0.27
State	-1.18	0.14	-0.14
Zip	-0.74	0.00	0.41
Password	-0.89	0.68	-0.41
<b>All fields (baseline)</b>	<b>90.99</b>	<b>98.52</b>	<b>97.85</b>

distinguishing between genuine and imposter inputs. To achieve this, we employed three different classifiers: the Decision Tree Classifier (DT), Support Vector Machines (SVM), and Multi-Layer Perceptron Classifier (MLP). Each classifier was trained separately using data from individual input fields, allowing us to assess how well each field performs.

The results of the classifiers with various input fields are summarized in Table 3. The MLP consistently demonstrated superior performance across most fields, achieving the highest accuracy of 95.56% for the email feature. This indicates that the typing patterns associated with email input are particularly distinct, likely due to users' familiarity with their email addresses. Similarly, the address and name features also performed well, with accuracies of 93.41% and 93.01%, respectively. These findings suggest that users exhibit stable typing behavior when entering these common data fields.

In contrast, the SVM classifier exhibited lower performance across all fields. The highest accuracy recorded was 91.67% for the address, with other fields such as state and password showing even lower accuracies of 70.43% and 86.01%, respectively. This suggests that the SVM may not capture the nuances of keystroke dynamics as effectively as the DT classifier, particularly in more complex input scenarios.

The DT classifier produced moderate results, with the email field achieving an accuracy of 91.37% and

the address field at 90.32%. However, the DT classifier also struggled with fields like state, where it only reached an accuracy of 79.57%. This variability in performance indicates that while the Decision Tree has the capability to learn complex patterns, it may require further tuning or additional training data to fully utilize the keystroke dynamics captured.

The analysis reveals significant differences in accuracy among the various input fields, reflecting the distinct typing behaviors associated with each. For instance, the high accuracy of the email field underscores the reliability of keystroke dynamics in identifying consistent patterns. Conversely, the lower accuracy for fields such as password and state suggests greater variability in user behavior, which could complicate the identification process.

Furthermore, the comparative performance of the classifiers highlights the importance of selecting the appropriate algorithm for keystroke dynamics. The DT classifier emerged as the most effective option, particularly for fields where users tend to type consistently. The SVM's lower accuracy indicates potential limitations in its applicability to this specific dataset, while the MLP's moderate performance suggests that further refinement may enhance its capabilities.

Overall, the findings from this experiment emphasize the necessity of considering both the choice of features and the classifier used in effectively leveraging keystroke dynamics for user authentication. The variability in performance across different input fields and classifiers will inform subsequent experiments, particularly those exploring feature removal and optimization strategies to improve classification accuracy.

#### 4.3.2 Field Removal Analysis

In the second experiment, we aimed to investigate the impact of removing individual keystroke fields on the classification accuracy of distinguishing between genuine and imposter inputs. By comparing the performance of each classifier with the combination of all fields to that with specific one removed, we sought to identify which field is most critical for accurate classification and which have a minimal impact.

The accuracy changes in each classifier when specific fields were removed are summarized in Table 4. For DT, the removal of certain fields resulted in varying degrees of accuracy change. Notably, the removal of the name field led to a decrease of 1.48%, while omitting the password field resulted in a 0.89% drop. In contrast, removing the email field, which was one of the strongest fields, resulted in only a minor increase of 1.48%. This suggests that while email is a strong predictor, its absence does not drastically hinder performance, likely due to the presence of other

contributing fields.

In addition to address, the most significant drop in accuracy was observed when the state field was removed, resulting in a decrease of 1.18%. This indicates that name input behavior has a substantial influence on classification. Conversely, the removal of country field did not lead to any negative impact on accuracy, which means that compared to other fields, most people are more familiar and coherent with the input of national fields, and the information that this field may provide is not as obvious.

For the SVM classifier, the results were also insightful. The removal of the name field caused a significant increase in accuracy by 0.69%, which indicates that the SVM struggled to capture relevant patterns associated with name inputs. Conversely, the removal of the Zip field resulted in an increase of 0.41%, which is unexpected, suggesting that the SVM may rely less on this field in its overall classification strategy. This contrasts sharply with the Decision Tree results, highlighting the differences in how the classifiers utilize specific fields.

The MLP classifier showed a different trend, with the removal of the name field leading to an increase in accuracy of 0.69%. The MLP appears to struggle with accurately classifying name inputs, similar to the behavior observed in the SVM. However, the MLP's reliance on the email field showed a decrease of 0.27%, suggesting that it may not be as reliant on this particular field as the DT classifier.

Overall, the analysis indicates that different fields contribute unequally to the classification accuracy across various classifiers. The DT classifier remains sensitive to the removal of fields like address and name, while the SVM and MLP exhibit less sensitivity, demonstrating their different underlying mechanisms for handling input data.

These results emphasize the importance of field selection in keystroke dynamics analysis. fields such as email and address are shown to be crucial for maintaining classification accuracy in Decision Tree models, while the SVM and MLP classifiers display varied reliance on fields, indicating that the optimization of field sets can lead to improved performance. The results of this experiment will provide ideas for the next section of the experiment, aimed at improving field selection strategies to enhance the efficiency of keystroke dynamics in user authentication systems.

#### 4.3.3 Optimal Field Selection Using Genetic Algorithm

In the third experiment, we employed Genetic Algorithm (GA) to identify the optimal combination of input fields that maximizes classification accuracy (Ji

Table 5: Top-5 Field-Combinations (DT).

Name	Email	Phone Number	Country	Address	City	State	ZIP	Password	Accuracy
×	✓	×	✓	✓	×	×	×	✓	92.88
×	✓	×	✓	✓	×	✓	×	✓	91.40
✓	✓	×	✓	✓	×	×	×	✓	90.59
×	×	✓	✓	✓	×	×	×	✓	90.59
×	✓	×	✓	×	✓	×	×	✓	90.43

Table 6: Top-5 Field-Combinations (SVM).

Name	Email	Phone Number	Country	Address	City	State	ZIP	Password	Accuracy
✓	✓	✓	✓	✓	✓	✓	✓	×	99.19
✓	✓	✓	✓	✓	✓	×	✓	×	98.92
×	✓	✓	✓	✓	✓	✓	✓	×	98.79
✓	✓	✓	✓	×	×	✓	✓	×	98.66
✓	×	×	✓	✓	✓	✓	✓	×	97.45

Table 7: Top-5 Field-Combinations (MLP).

Name	Email	Phone Number	Country	Address	City	State	ZIP	Password	Accuracy
×	✓	✓	✓	✓	✓	✓	✓	×	98.52
×	✓	✓	✓	✓	✓	×	✓	✓	98.38
✓	✓	✓	✓	✓	✓	✓	✓	✓	97.84
×	✓	✓	✓	×	×	✓	✓	✓	97.18
×	×	×	✓	✓	✓	✓	✓	✓	96.23

et al., 2021). GA are optimization search techniques inspired by the principles of natural selection and genetics. The fundamental idea is to mimic the evolutionary process in nature, utilizing operations such as selection, crossover, and mutation to progressively improve solutions. GA are particularly effective in solving complex optimization problems, including field selection, path planning, and various machine learning applications. The steps of GA include:

1. **Initialization of Population:** Randomly generate a set of candidate solutions (individuals), each represented as a binary array where 1 indicates field inclusion and 0 indicates exclusion.
2. **Fitness Evaluation:** Assess the performance of each individual using a defined fitness function, which in this study is the classification accuracy.
3. **Selection:** Select individuals based on their fitness, favoring those with higher fitness scores.
4. **Crossover:** Perform crossover operations on selected individuals to create new offspring.
5. **Mutation:** Apply mutations to the offspring with a certain probability to maintain population diversity.
6. **Replacement:** Replace part of the old population with new offspring to form the next generation.

7. **Iteration:** Repeat the above steps until a termination condition is met (e.g., reaching a specified number of generations or a fitness threshold).

In this experiment, the parameters for the Genetic Algorithm are set as follows:

- **Population Size:** 50
- **Number of Generations:** 50
- **Mutation Rate:** 0.05

During the execution of the GA, several individuals (feature combinations) were generated, evaluated, and evolved over multiple generations. Each individual was represented as a binary array, where a value of 1 indicates the inclusion of a specific field and a value of 0 indicates its exclusion. The results of the genetic algorithm under three classifiers are shown in Tables 5, 6, 7, indicating the best individuals and their respective fitness scores (classification accuracy).

For the DT classifier, the top-performing field combinations predominantly included the Address, Email, and Country fields, achieving a maximum accuracy of 92.88%. The presence of these fields consistently contributed to better performance, highlighting their importance in decision-making processes within this algorithm.

In contrast, the SVM classifier demonstrated a significant improvement in accuracy, with a peak of 99.19%. This classifier consistently utilized a wider

range of fields, including Address, City, Email, Country, and Name. The ability of SVM to leverage these fields suggests that it may benefit from the enhanced representation of data provided by these specific attributes, thus improving performance.

The MLP results also indicate high accuracy, with the best combination reaching 98.52%. Similar to SVM, MLP favored a combination of multiple fields, particularly those related to the Address, City, and Email, while also showing sensitivity to the inclusion of Phone Number and State.

The field selection results reveal intriguing patterns among the classifiers. Notably DT did not select any combinations that included the ZIP field, which may suggest that the geographical granularity provided by ZIP codes did not enhance the decision-making process for this particular model. This could be attributed to DT's reliance on more categorical and high-level fields like Address and Country, which effectively capture the necessary information for classification without the need for finer detail. In contrast, both SVM and MLP frequently included the ZIP field in their top combinations, achieving accuracies of up to 99.19% and 98.52%, respectively. This indicates that these classifiers can leverage the additional detail provided by ZIP codes to improve their predictive performance. The inclusion of ZIP in SVM and MLP may enhance the model's ability to distinguish between subtle variations in data, which is particularly useful in complex datasets.

Another noteworthy observation is that SVM consistently excluded the Password field across all selected combinations. This could imply that the Password attribute did not contribute significantly to the classification task, possibly due to its highly sensitive and varied nature, which might not offer relevant predictive power in this context. Conversely, the MLP models included the Password field in some combinations, suggesting that it might be beneficial in specific scenarios, though its contribution was less prominent compared to other fields.

Overall, the results from this experiment underscore the effectiveness of the Genetic Algorithm in optimizing field (field) selection for keystroke dynamics analysis. By identifying the most relevant field, we can improve the classification accuracy of identifying genuine and imposter inputs. This experiment also highlights the importance of careful field selection in machine learning applications, suggesting that certain fields provide significant predictive power while others may not contribute as effectively to the overall classification task.

## 5 CONCLUSION

In this study, we explored the application of keystroke dynamics as a behavioral biometric for distinguishing between genuine and imposter user inputs across multiple common e-Commerce web services. Our work aimed to enhance the security of online systems by leveraging the unique typing patterns of users, addressing the critical challenge of ensuring reliable user authentication.

We developed a comprehensive keystroke data collection system utilizing the Django framework, enabling us to capture and analyze typing behavior in real-time across three distinct web services: an air ticket service, an online shopping system, and a car rental service. Through extensive data collection, each user provided both authentic and imposter inputs, resulting in a balanced dataset that allowed for a robust analysis of keystroke dynamics.

In the experiment, we explored the differences in input fields and observed changes in classification accuracy by adding or removing fields. In addition, genetic algorithms are used to find the field combination set with the highest classification accuracy among all classical input fields. This is of great help for improving model performance through field selection in the future, and also provides a foundation for further research in this field.

## ACKNOWLEDGMENTS

This work was supported by NSF Award TI-2122746.

## REFERENCES

- Ahmed, A. A. and Traore, I. (2013). Biometric recognition based on free-text keystroke dynamics. *IEEE transactions on cybernetics*, 44(4):458–472.
- Dowland, P. S. and Furnell, S. M. (2004). A long-term trial of keystroke profiling using digraph, trigraph and keyword latencies. In Deswarte, Y., Cuppens, F., Jajodia, S., and Wang, L., editors, *Security and Protection in Information Processing Systems*, pages 275–289, Boston, MA. Springer US.
- Gunetti, D. and Picardi, C. (2005). Keystroke analysis of free text. *ACM Trans. Inf. Syst. Secur.*, 8(3):312–347.
- Ji, J.-J., Guo, Y.-N., Gao, X.-Z., Gong, D.-W., and Wang, Y.-P. (2021). Q-learning-based hyperheuristic evolutionary algorithm for dynamic task allocation of crowdsensing. *IEEE Transactions on Cybernetics*, 53(4):2211–2224.
- Killourhy, K. S. and Maxion, R. A. (2009). Comparing anomaly-detection algorithms for keystroke dynamics. In *2009 IEEE/IFIP international conference*



- on dependable systems & networks*, pages 125–134. IEEE.
- Messerman, A., Mustafić, T., Camtepe, S. A., and Albayrak, S. (2011). Continuous and non-intrusive identity verification in real-time environments based on free-text keystroke dynamics. In *2011 International Joint Conference on Biometrics (IJCB)*, pages 1–8. IEEE.
- Monaco, J. V., Bakelman, N., Cha, S.-H., and Tappert, C. C. (2012). Developing a keystroke biometric system for continual authentication of computer users. In *2012 European Intelligence and Security Informatics Conference*, pages 210–216. IEEE.
- Sun, Y., Ceker, H., and Upadhyaya, S. (2016). Shared keystroke dataset for continuous authentication. In *2016 IEEE international workshop on information forensics and security (WIFS)*, pages 1–6. IEEE.
- Vural, E., Huang, J., Hou, D., and Schuckers, S. (2014). Shared research dataset to support development of keystroke authentication. In *IEEE International joint conference on biometrics*, pages 1–8. IEEE.
- Wahab, A. A., Hou, D., and Schuckers, S. (2023). A user study of keystroke dynamics as second factor in web MFA. In Shehab, M., Fernández, M., and Li, N., editors, *CODASPY'2023*, pages 61–72. ACM.



SCITEPRESS  
SCIENCE AND TECHNOLOGY PUBLICATIONS