

Authentication Using Keystrokes Dynamic For Static Text With Hybrid Deep Learning Technique

Namisha Bhasin^{1*}, Sanjay Kumar Sharma², Rajesh Mishra³

^{1,2,3}Gautam Buddha University, Greater Noida, India

Abstract

The Behavioral authentication via keystroke dynamics provides a non-intrusive and hardware-independent means of identity verification. However, prior deep-learning approaches often suffer from class imbalance and limited generalization. This study proposes a hybrid deep-learning model combining multi-head feature fusion, Conv1D layers, capsule networks, bidirectional LSTM, attention mechanisms, and Monte Carlo dropout for static-text authentication. To mitigate imbalance in the CMU Keystroke Dynamics dataset, oversampling and class-weighted loss were employed. The model was trained for 20 epochs and evaluated through five-fold cross-validation. Experimental results yield an accuracy of 99.3%, outperforming recent baselines. The proposed framework effectively captures spatial temporal dependencies and quantifies uncertainty, contributing a robust and interpretable solution for real-time behavioral authentication.

Keywords: Keystroke Dynamics, Deep Learning, BiLSTM, Capsule Network, Feature Fusion.

Introduction

In recent years, the spread of intelligent systems powered by the machine learning (ML) and the deep learning (DL) have transformed the landscape of behavioural biometrics, especially in the authentication and in order to access the control systems. The various behavioural biometric traits, such as keystroke dynamics analyze an individual's typing rhythm that gained traction due to the non-intrusiveness, cost-effectiveness, and suitability for continuous authentication scenarios. Specifically, the fixed-text keystroke dynamics in the users recurrent type a predefined password or phrase that emerged as the practical and controlled method for modelling the user-specific typing behaviour.

The exponential growth of digital systems and smart environments are including the secure banking, enterprise access systems, and the remote education platforms that intensified the demand for robust and a user-friendly authentication technique [1]. The traditional approaches, like the passwords and tokens are vulnerable to theft, phishing, and the replay attacks in which the biometric techniques like fingerprints and the facial recognition are often require additional hardware and are raise the privacy concerns. In contrast, the keystroke dynamics provides a software-based behavioural biometric solution that works seamlessly on any keyboard-enabled device [2-6].

The increasing complexity and variability of user input data, Artificial Intelligence (AI) are become essential in the extraction of meaningful patterns from the raw keystroke timing features. Particularly, the ML and the DL models are shown great promise in capturing the individual typing signatures, even when the data is noisy or collected over time [7]. In the fixed-text scenarios, the timing features, like the hold time and flight time are leveraged in order to build the robust user profiles for authentication purposes [8-10].

Classical ML models that are effective in some cases depend heavily on the handcrafted features and struggle with generalization across large user bases or varied typing conditions. To address these limitations, the DL architectures, like the "Convolutional Neural Networks (CNNs)" and "Long Short-Term Memory (LSTM)" networks are extensively adopted for modelling the temporal and spatial characteristics of the keystroke dynamics data [11-12]. However, the CNNs are typically rely on pooling operations that are discard the spatial relationships between features. To overcome this, the concept of the Capsule Networks (CapsNets), are introduced by the Sabour et al.,

(2017) [13], that proposed as a more spatially aware alternative. The capsules use dynamic routing in order to maintain the part-whole relationships and are shown to improve the generalization in pattern recognition tasks where a structural information is important [14]. This capability makes the CapsNets particularly suitable for keystroke dynamics, where the temporal alignment and positional context of keystrokes are critical. A key limitation in the current literature is the class imbalance problem, where genuine-user samples vastly outnumber impostor samples, leading to biased models that favor the majority class. Many prior studies have reported deceptively high accuracies while failing to detect impostors effectively an issue that directly impacts Equal Error Rate (EER) and real-world reliability. Furthermore, interpretability and uncertainty quantification remain underexplored, despite their importance in safety-critical authentication systems. In this study, we propose a hybrid DL model that integrates the CNN, BiLSTM, along with the Capsule Networks, and an attention mechanism, tailored especially for the fixed-text keystroke dynamics authentication. The architecture is trained and tested on the widely used CMU keystroke dataset that captures detailed typing behavior from multiple users. We present a comprehensive evaluation based on accuracy, precision, recall, and F1-score, along with visual diagrams, like confusion matrices and performance curves [15-19].

The key contributions of this paper are as follows:

- A novel hybrid architecture for fixed-text keystroke authentication that fuses Conv1D, Capsule, BiLSTM, and Attention layers for richer representation learning.
- An improved training strategy employing rebalancing and uncertainty estimation through Monte Carlo dropout.
- A comprehensive experimental evaluation on the CMU Keystroke Dynamics dataset, reporting balanced accuracy, EER, and standard deviation over 10-fold validation.
- An analysis of model interpretability, convergence behavior, and limitations with respect to language and platform variability.

This research contributes to the broader field of pattern recognition by demonstrating how multi-level feature integration and balance-aware optimization can yield more robust and interpretable behavioral authentication systems.

The study contributes in the area of behavioural authentication by representing how the advanced DL techniques are effectively model the user-specific typing behavior in the fixed-text settings with offering scalability, interpretability, and resistance to the spoofing.

Related Work

In this section explored the various studies done by previous researchers in the area of the keystroke dynamics and behavioural authentication.

Wang et al., (2024) [20] explored the blend of SMOTE oversampling with ensemble classifiers, such as, XGBoost, Random Forest, and Bagging. Evaluated on the two different CMU and Django-collected datasets, the author significantly improves the recognition of minority-class impostors for boosting overall F1-score and recall.

Wyciślik et al., (2024) [21] presented the novel neural architectures that tailored for the variable-text keystroke on the basis of the user identification. The benchmarked against multiple open datasets are proposed by the author that demonstrates an improved accuracy and robustness over the classical techniques.

The author of the Risto & Graven, (2024) [22] introduced and validated the new controlled dataset covering six different fixed-text passwords. Further, the author investigated the effects of password complexity and the length on Equal Error Rate (EER). As per the outcomes, the true accept/reject performance, provides the practical benchmark for future static-text authentication research.

The author of Renuka & Suresh, (2024) [23] presented the integrated approach that fuses keystroke and the fingerprint biometrics. The DL models capture the typing patterns via

CNN and BiLSTM, and robustness is further enhanced with the help of the image-based fingerprint analysis that achieve the high security in a multi-modal setup.

The Author of Teotia et al., (2023) [24] provided the comprehensive exploration of the various ML based algorithms (e.g., Support Vector Machine (SVM), Random Forest), feature selection techniques, and the preprocessing of pipelines. The authors compared the different algorithmic performance under the varied conditions that highlight the trade-offs in feature engineering and classifier choice.

The author of Stragapede et al., (2024) [25] proposed the transforming PIN keystroke sequences into images that capture the temporal dynamics for a passive mobile authentication. The Image-based encoding enables the CNN models in order to achieve the low EER of 6.7%, outperforming the previous DL methods on a custom dataset for 17 users. The Clarkson University and the RIT developed the blend of SMOTE oversampling with ensemble models (Random Forest, XGBoost). Evaluated on the benchmark dataset, the method significantly improves the detection of a minority-class impostors that achieved the stronger recall, precision and F1 compared to the classical ML algorithms [26].

AbdelRaouf et al., (2023) [27] proposes an optimized CNN architecture tailored for the keystroke features. Evaluated on standard benchmarks, the model achieves a substantial improvement in the classification accuracy over the traditional ML approaches that demonstrate the well-crafted CNNs are effective for the typing-based biometric authentication. The article in 2023 demonstrates a fusion model using the gestures touch and the keystroke dynamics in order to resolve the usability issues. DL architectures (LSTM and MLP) are employed, significantly for enhancing the continuity and robustness in mobile authentication scenarios [28].

The author of [29] introduced the first transformer model for a free-text keystroke authentication. The model achieves an EER of the 3.84% that outperforms the CNN and RNN baselines, showcasing the potential of attention mechanisms for the temporal typing patterns. The BehaveFormer developed by Senerath et al., (2023) [30] integrated the keystroke dynamics with the smartphone sensor data (accelerometer, gyroscope). The dual attention transformers reach the 2–3% EER on the multiple datasets that demonstrate the combination of the behavioural typing with the motion data significantly enhances the continuous authentication accuracy.

Table 1. Comparative Literature of Review

Study	Approach	Dataset(s)	Notable Metrics
Wang et al., (2024) [20]	Ensemble + SMOTE	CMU + Django data	Improved precision/recall on impostor class
Wyciślik et al., (2024) [21]	Deep learning models	Public benchmarks	Higher accuracy over traditional methods
Risto & Graven, 2024 [22]	New fixed-text dataset + analysis	Custom six-password dataset	EER: 10.2–18.1%, TAR/TRR performance
Renuka & Suresh, 2024 [23]	CNN-BiLSTM fused with fingerprint	Dual-modal keystroke + fingerprint	High authentication performance
Teotia et al., (2023) [24]	ML classifier survey	Multiple datasets	Comparative performance across ML algorithms

Study	Approach	Dataset(s)	Notable Metrics
Stragapede et al., (2024) [25]	Image encoding + CNN	Custom mobile PIN dataset	EER: 6.7%, TAR: 94.4%
AbdelRaouf et al., (2023) [27]	Optimized CNN	Public keystroke datasets	Accuracy > traditional ML
Stragapede et al., (2023) [29]	Transformer architecture	Aalto mobile keystroke DB	EER: 3.84%
Senerath et al., (2023) [30]	Keystroke + IMU + Transformer	Aalto, HuMldb, HMOG	EER: 1.8–2.95%

Background and Motivation

The convergence of the AI, and authentication technologies are transformed the traditional systems into intelligent, real-time solutions. The proliferation of a smart wearable devices and the sensor-enabled equipment are the ability in order to collect the continuous physiological data that enables the early detection of diseases and proactive interventions. However, the technological shift also introduces the several complexities related to data privacy, security, heterogeneity, and the model generalization across a decentralized environment [31].

Particularly in the domain, that maintained the integrity and confidentiality of sensitive records is paramount. As data is collected and transmitted across networks, it becomes susceptible to the cyber-attacks and the data breaches. Hence, robust privacy-preserving protocols and learning mechanisms are crucial in order to ensure the trust and compliance with the regulations [32].

The rising volume and velocity of data also presented a challenge for the centralized ML models that often requires aggregating data in the central repository. Such systems are not only prone to the privacy risks but also the face bottlenecks in scalability and latency. This led to the adoption of DL frameworks that allows the decentralized model training while keeping the data by leveraging different layers [33].

To enhance the security of local updates and the privacy of exchanged model parameters, the various techniques are gained prominence. These technologies enables the computation on encrypted data that provides an extra layer of protection during the system training and communication. Additionally, as explored by the author of Chang et al., (2022) [34], the integration of ML for behaviour-based data analysis, like keystroke dynamics are demonstrates that the AI are effectively handle sensitive, unstructured data in privacy-sensitive applications, further emphasizing the need for robust ML-driven cyber-protection in distributed healthcare systems.

This paper builds upon these motivations and proposes a secure, privacy-aware, and efficient DL-based framework for keystroke dynamics. The proposed model not only ensures data confidentiality but also achieves high predictive performance in real-time anomaly detection.

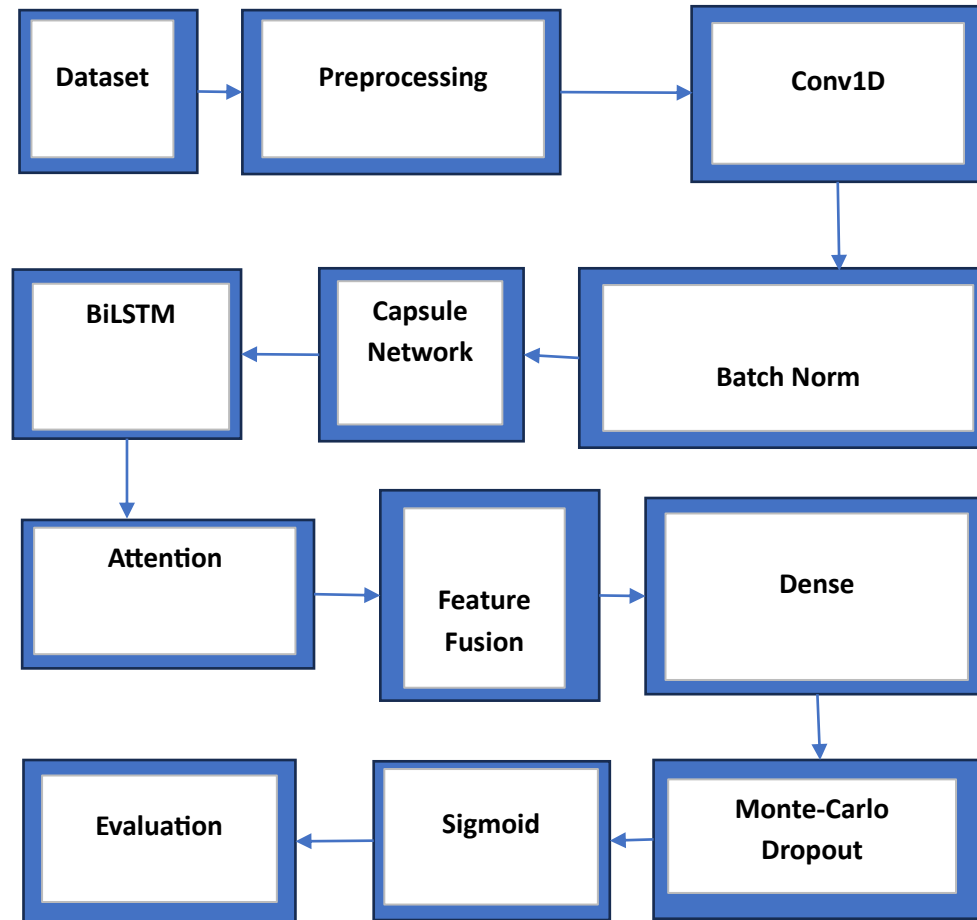
Research Methodology

The proposed keystroke authentication framework follows the architectural pipeline illustrated in Figure 1. The methodology is beginning from dataset preprocessing and progressing through multi-stage deep feature learning, classification, and evaluation.

3.1 Dataset Description

This study utilizes the publicly available CMU Keystroke Dynamics Dataset, a benchmark behavioural biometric dataset widely used for fixed-text authentication research. The dataset consists of 51 subjects, each contributing 8 sessions with 50 repetitions per session, resulting in a total of 20,400 keystroke samples. Each sample corresponds to the typing of a predefined password string “tie5Roanl,” ensuring consistency across users and sessions. The dataset captures fine-grained temporal characteristics of typing behaviour rather than lexical content, making it suitable for user authentication based on neuromotor patterns.

Figure 1: system-level workflow



Each keystroke sequence is represented as a multivariate temporal vector composed of timing-based features. Specifically, the dataset includes hold time (H), which measures the duration between key press and key release; down-down latency (DD), which represents the interval between successive key press events; and up-down latency (UD), which captures the delay between key release and the next key press. Let a single raw keystroke instance be defined as given in equation 1.

$$X_{raw} = \{x_1, x_2, \dots, x_T\}, \quad x_i \in \mathbb{R}^d \quad \dots (1)$$

where T denotes the number of keystroke-related timing events in the password sequence and d represents the dimensionality of the timing feature vector. Thus, each typing sample forms a structured time-series representation of behavioral dynamics.

To ensure numerical stability and mitigate inter-feature scale differences, each timing feature is standardized using Z-score normalization:

$$x' = \frac{x - \mu}{\sigma} \quad \dots (2)$$

where μ and σ denote the mean and standard deviation of the corresponding feature across the training data. After normalization, the dataset is reshaped into a tensor suitable for deep learning architectures (see equation 3).

$$X \in \mathbb{R}^{N \times C \times T} \quad \dots (3)$$

where N is the number of samples, $C = 1$ represents the input channel, and T corresponds to the temporal sequence length derived from the password typing events.

For authentication purposes, the problem is formulated as a user-dependent binary classification task. Given a selected target user, samples belonging to that user are labeled as genuine, while all other users' samples are treated as impostors. The binary label assignment is defined as given in equation 4.

$$y = \begin{cases} 1, & \text{if the sample belongs to the genuine} \\ 0, & \text{if the sample belongs to an imposter} \end{cases} \quad \dots (4)$$

Let N_{pos} denote the number of genuine samples and N_{neg} denote the number of impostor samples. The total number of samples is therefore given in equation 5.

$$N = N_{pos} + N_{neg} \quad \dots (5)$$

Because biometric datasets inherently exhibit class imbalance in user-dependent setups (where impostor samples significantly outnumber genuine samples), class-weighted learning is adopted to ensure balanced optimization. The positive and negative class weights are computed as depicted in equation 6.

$$w_{pos} = \frac{N}{2N_{pos}}, \quad w_{neg} = \frac{N}{2N_{neg}} \quad \dots (6)$$

These weights are later incorporated into the loss formulation to prevent model bias toward the majority class.

Overall, the CMU Keystroke Dynamics Dataset provides a structured, session-based, fixed-text behavioural biometric dataset characterized by high intra-user consistency and inter-user variability. Its multivariate temporal structure, repeated session recordings, and balanced experimental protocol make it well-suited for evaluating deep learning models designed for keystroke-based user authentication.

3.2 Data Preprocessing

As depicted in Figure 1, the data preprocessing stage transforms the normalized keystroke timing vectors described in Section 3.1 into structured representations suitable for deep hierarchical learning. While Section 3.1 established the dataset formulation, normalization procedure, and binary labeling scheme, this section focuses on the transformation, restructuring, and preparation steps that enable effective temporal feature extraction and stable model optimization.

After normalization, each keystroke sample is reshaped into a channel-first tensor format compatible with one-dimensional convolution operations. The structured input tensor is defined as in equation 3.

To ensure robust generalization and unbiased evaluation, stratified cross-validation is applied during dataset partitioning. The complete dataset D is divided into $K = 5$ mutually exclusive folds such that the equation in 7.

$$D = \bigcup_{k=1}^K D_k, \quad D_i \cap D_j = \emptyset \text{ for } i \neq j \quad \dots (7)$$

Each fold maintains approximately equal proportions of genuine and impostor samples, preserving class distribution consistency across training and validation splits. During each iteration, four folds are used for training and one fold for validation, ensuring that every sample is evaluated exactly once.

Given the inherent class imbalance in user-dependent biometric authentication, weighted optimization is incorporated during training. Although class weights are defined in Section 3.1, their practical application occurs during preprocessing for model training. The weighted binary cross-entropy objective is expressed as given in equation 8.

$$\mathcal{L}_{BCE} = -w_{pos}y\log(\hat{y}) - w_{neg}(1 - y)\log(1 - \hat{y}) \quad \dots (8)$$

where w_{pos} and w_{neg} are precomputed class-balancing coefficients and $\hat{y}$ denotes the predicted probability. This formulation ensures that minority-class genuine samples contribute proportionally to gradient updates, thereby preventing bias toward the dominant impostor class.

Furthermore, mini-batch training is employed to improve computational efficiency and stabilize gradient descent. Let a mini-batch of size B be denoted as depicted in equation 9.

$$\mathcal{B} = \{(X_i, y_i)\}_{i=1}^B \quad \dots (9)$$

The empirical batch loss minimized during optimization is given in equation 10.

$$\mathcal{L}_{batch} = \frac{1}{B} \sum_{i=1}^B \mathcal{L}_{BCE}^{(i)} \quad \dots (10)$$

where $\mathcal{L}_{BCE}^{(i)}$ corresponds to the weighted loss of the i - th sample in the batch.

The preprocessing stage in Figure 1 ensures that the normalized keystroke sequences are properly reshaped, partitioned through stratified cross-validation, balanced through weighted optimization, and organized into mini-batches for efficient training. These structured preparations enable stable convergence of the subsequent convolutional, capsule, recurrent, and attention-based components of the proposed authentication framework.

3.3 Multi-Head Feature Fusion

As illustrated in Figure 1, the proposed framework incorporates a multi-head feature fusion strategy to enhance the discriminative representation of keystroke dynamics prior to deep hierarchical modeling. Keystroke authentication relies on subtle neuromotor patterns that manifest across different temporal perspectives, including raw timing sequences, transition-based digraph statistics, and positional encoding of key order. A single representation may fail to capture the complementary information embedded in these heterogeneous temporal descriptors. Therefore, multiple feature streams are processed in parallel and fused to form a unified representation. The multi-head design decomposes this representation into three complementary feature mappings:

1. Raw timing representation F_1 ,
2. Digraph latency representation F_2 ,
3. Positional order encoding representation F_3 .

Each head applies an independent learnable transformation function $\varphi_i(\cdot)$, defined as:

$$F_i = \varphi_i(X), \quad i \in \{1,2,3\} \quad \dots (11)$$

where φ_i represents a learnable mapping implemented through parallel Conv1D filters. For each head, convolutional feature extraction can be expressed as given in equation 12.

$$F_i(t) = \sigma(W_i * X(t) + b_i) \quad \dots (12)$$

where:

- W_i denotes the convolutional kernel weights of the i - th head,
- $*$ represents the 1D convolution operation,

- b_i is the bias term,
- $\sigma(\cdot)$ is a nonlinear activation function (ReLU).

This parallel structure allows each head to specialize in learning distinct local temporal correlations within the keystroke sequence. The raw timing head emphasizes absolute hold and flight durations, the digraph head captures transition dynamics between adjacent keys, and the positional encoding head reinforces sequential order dependencies.

Following independent feature extraction, the resulting feature maps are fused through concatenation along the feature dimension:

$$F_{fusion} = [F_1 || F_2 || F_3] \quad \dots (13)$$

where $||$ denotes concatenation. If each feature stream produces a representation in $\mathbb{R}^{T \times d_i}$, then the fused representation becomes as given in equation 14.

$$F_{fusion} \in \mathbb{R}^{T \times (d_1 + d_2 + d_3)} \quad \dots (14)$$

This fusion strategy preserves complementary temporal characteristics while increasing representational capacity. Unlike simple averaging, concatenation maintains the distinct semantic contributions of each head, enabling subsequent layers to learn weighted combinations of heterogeneous behavioral cues.

To prevent dominance of any individual feature stream and encourage stable optimization, batch normalization is applied after concatenation:

$$\hat{F}_{fusion} = \frac{F_{fusion} - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} \quad \dots (15)$$

where μ_B and σ_B^2 are batch statistics computed over mini-batches. This normalization ensures consistent scale across feature channels and facilitates deeper nonlinear transformations in later stages.

The multi-head feature fusion mechanism depicted in Figure 1 thus acts as a high-capacity representation learner that integrates diverse keystroke descriptors into a unified embedding. By preserving multiple temporal perspectives before feeding into the convolutional–capsule–recurrent hierarchy, the model enhances its ability to distinguish genuine users from impostors based on subtle behavioral micro-patterns embedded within fixed-text typing sequences.

3.4 Temporal Feature Extraction via Conv1D

As illustrated in Figure 1, temporal feature extraction is performed using one-dimensional convolutional layers to capture short-range dependencies and local temporal dynamics embedded within the keystroke timing sequences. Keystroke authentication relies heavily on micro-patterns such as key hold durations and inter-key latencies, which occur sequentially across the typing process. Conv1D operations are particularly suitable for this structured sequential input because they learn translation-invariant filters that detect local timing motifs irrespective of their exact position within the sequence.

Let the fused feature representation obtained from the multi-head fusion stage be denoted as in equation 16.

$$F_{fusion} = \{f_1, f_2, \dots, f_T\}, \quad f_t \in \mathbb{R}^{d_f} \quad \dots (16)$$

where T represents the sequence length and d_f is the dimensionality of the fused feature vector at time step t . A one-dimensional convolution operation with kernel size k applies a sliding window across the temporal axis. For the j -th filter, the convolutional output at position t is computed (see equation 17).

$$z_t^{(j)} = \sum_{i=0}^{k-1} W_i^{(j)} \cdot f_{t+i} + b^{(j)} \quad \dots (17)$$

where:

- $W^{(j)} \in \mathbb{R}^{k \times d_f}$ denotes the learnable convolutional weights,
- $b^{(j)}$ is the bias term,
- k is the receptive field size.

The convolution operation aggregates information from neighboring time steps, enabling the network to learn local timing correlations between consecutive key events. After convolution, a nonlinear activation function is applied using equation 18.

$$h_t^{(j)} = \sigma(z_t^{(j)}) \quad \dots (18)$$

where $\sigma(\cdot)$ is the Rectified Linear Unit (ReLU), defined as given in equation 19.

$$\sigma(x) = \max(0, x) \quad \dots (19)$$

This nonlinearity introduces sparsity and enhances discriminative capacity by suppressing negative activations.

When multiple filters are used, the output feature map becomes as given in equation 20.

$$H_t = [h_t^{(1)}, h_t^{(2)}, \dots, h_t^{(M)}] \quad \dots (20)$$

where M is the total number of convolutional filters. Consequently, the convolutional layer transforms the input into a higher-dimensional temporal representation (see equation 21).

$$H \in \mathbb{R}^{T' \times M} \quad \dots (21)$$

where T' depends on kernel size and padding configuration.

To enhance hierarchical abstraction, multiple Conv1D layers may be stacked sequentially. For layer l , the transformation can be generalized (see equation 22).

$$H^{(l)} = \sigma(W^l * H^{(l-1)} + b^{(l)}) \quad \dots (22)$$

This layered architecture allows the model to progressively learn low-level timing transitions in early layers and higher-order behavioral motifs in deeper layers, as depicted in Figure 1.

To improve generalization and prevent overfitting, dropout regularization is incorporated after convolution (see equation 23).

$$\tilde{H}^{(l)} = H^{(l)} \odot M^{(l)} \quad \dots (23)$$

where $M^{(l)}$ is a Bernoulli mask and $\odot$ denotes element-wise multiplication. This stochastic regularization forces the network to rely on distributed feature representations rather than isolated neurons.

Through this Conv1D-based temporal encoding mechanism shown in Figure 1, the model effectively captures localized keystroke timing dependencies and extracts discriminative sequential features. These learned temporal embeddings serve as foundational inputs for subsequent capsule, recurrent, and attention modules, enabling robust differentiation between genuine users and impostors based on fine-grained typing behavior patterns.

3.5 Capsule Network with Dynamic Routing

As illustrated in Figure 1, the convolutional feature maps extracted from the Conv1D layers are further processed using a Capsule Network with dynamic routing to preserve hierarchical relationships and part-whole dependencies in keystroke dynamics. While conventional convolutional layers capture local temporal patterns effectively, they may lose spatial-temporal orientation information due to pooling and scalar activations. Capsule Networks address this limitation by representing features as vectors rather than scalars, where both magnitude and orientation encode meaningful information about behavioral patterns.

Let the output of the final Conv1D layer be denoted as in equation 21. These scalar activations are reorganized into primary capsules. Each capsule is a vector representation defined as in equation 23.

$$u_i \in \mathbb{R}^{d_c} \quad \dots (23)$$

where d_c is the capsule dimensionality and i indexes individual capsules. Unlike traditional neurons that output scalars, capsules output vectors whose lengths indicate the probability of feature presence, while orientations encode instantiation parameters of the detected pattern.

Each lower-level capsule u_i predicts the output of a higher-level capsule v_j through a linear transformation (see equation 24).

$$\hat{u}_{j|i} = W_{ij}u_i \quad \dots (24)$$

where $W_{ij} \in \mathbb{R}^{d_v \times d_c}$ is a learnable weight matrix and d_v is the dimensionality of the higher-level capsule.

Dynamic routing determines how strongly each lower-level capsule contributes to higher-level capsules. The total input to capsule j is computed as a weighted sum of predictions (see equation 25).

$$s_j = \sum_i c_{ij} \hat{u}_{j|i} \quad \dots (25)$$

where c_{ij} are routing coefficients satisfying (see equation 26)

$$c_{ij} = \frac{\exp(b_{ij})}{\sum_k \exp(b_{ik})} \quad \dots (26)$$

Here, b_{ij} are log prior probabilities initialized to zero and iteratively updated based on agreement between lower and higher capsules.

To ensure that capsule outputs remain bounded between 0 and 1 in magnitude, a nonlinear squashing function is applied (see equation 27).

$$v_j = \frac{\|s_j\|^2 s_j}{1 + \|s_j\|^2 + \epsilon} \quad \dots (27)$$

This squashing function ensures that short vectors shrink toward zero while long vectors approach unit length, thereby representing the probability of feature presence through vector magnitude.

During routing iterations, the agreement between prediction $\hat{u}_{j|i}$ and output v_j is computed using equation 28.

$$a_{ij} = \hat{u}_{j|i} \cdot v_j \quad \dots (28)$$

The routing logits are then updated as given in equation 29.

$$b_{ij} \leftarrow b_{ij} + a_{ij} \quad \dots (29)$$

This iterative refinement strengthens connections between capsules that agree and weakens those that do not, enabling hierarchical clustering of temporal features.

In the context of keystroke authentication, lower-level capsules capture local digraph timing motifs learned by Conv1D layers, while higher-level capsules represent more abstract behavioral patterns such as consistent typing rhythm structures characteristic of genuine users. By preserving vector-based relationships rather than scalar activations, the Capsule Network enhances robustness against small temporal distortions in typing behavior.

As shown in Figure 1, the capsule outputs serve as structured high-level embeddings that retain both magnitude-based confidence and orientation-based relational encoding. These enriched representations are subsequently forwarded to recurrent and attention-based modules for global temporal modeling, ultimately improving discrimination between genuine and impostor typing patterns in the biometric authentication task.

3.6 Bidirectional LSTM

As illustrated in Figure 1, the high-level capsule embeddings are subsequently processed using a Bidirectional Long Short-Term Memory (BiLSTM) network to model long-range

temporal dependencies across the entire keystroke sequence. While Conv1D layers capture localized timing transitions and Capsule Networks preserve hierarchical feature relationships, keystroke authentication also requires modeling sequential consistency over the full typing trajectory. Human typing behavior exhibits rhythm patterns that extend beyond immediate neighboring keys; therefore, recurrent architectures are necessary to capture contextual dependencies spanning the complete sequence. Let the output of the Capsule Network be represented as a sequential embedding (see equation 30).

$$V = \{v_1, v_2, \dots, v_{T''}\}, \quad v_t \in \mathbb{R}^{d_v} \quad \dots (30)$$

where T'' denotes the sequence length after capsule processing and d_v is the capsule output dimension. The LSTM cell at time step t maintains a hidden state h_t and a memory cell state c_t . The internal gating mechanisms regulate information flow through the network.

The forget gate determines how much previous memory is retained (see equation 31).

$$f_t = \sigma(W_f v_t + U_f h_{t-1} + b_f) \quad \dots (31)$$

The input gate controls how much new information is incorporated (see equation 32).

$$i_t = \sigma(W_i v_t + U_i h_{t-1} + b_i) \quad \dots (32)$$

The candidate memory content is computed using equation 33.

$$\tilde{c}_t = \tanh(W_c v_t + U_c h_{t-1} + b_c) \quad \dots (33)$$

The memory cell is updated using equation 34.

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \quad \dots (34)$$

Finally, the output gate determines the hidden state depicted in equation 35 and 36.

$$o_t = \sigma(W_o v_t + U_o h_{t-1} + b_o) \quad \dots (35)$$

$$h_t = o_t \odot \tanh(c_t) \quad \dots (36)$$

Here, $\sigma(\cdot)$ represents the sigmoid activation function, and $\odot$ denotes element-wise multiplication.

In the bidirectional configuration, two LSTM networks operate in opposite temporal directions. The forward LSTM processes the sequence from $t = 1$ to T'' , producing hidden states $\vec{h}_t$, while the backward LSTM processes the sequence in reverse, producing $\tilde{h}_t$. These are defined as given in equation 37 and 38.

$$\vec{h}_t = LSTM_f(v_t, \vec{h}_{t-1}) \quad \dots (37)$$

$$\tilde{h}_t = LSTM_b(v_t, \tilde{h}_{t+1}) \quad \dots (38)$$

The final bidirectional representation at each time step is obtained by concatenation (see equation 39).

$$h_t^{bi} = [\vec{h}_t || \tilde{h}_t] \quad \dots (39)$$

This bidirectional formulation enables the model to incorporate both past and future context when evaluating each keystroke event. In keystroke authentication, this is particularly important because typing rhythm at any key press may depend on both preceding and succeeding key transitions.

The resulting BiLSTM output sequence (see equation 40).

$$H^{bi} = \{h_1^{bi}, h_2^{bi}, \dots, h_{T''}^{bi}\} \quad \dots (40)$$

encodes global temporal dependencies and long-term behavioral consistency patterns characteristic of genuine users. As depicted in Figure 1, this enriched sequential representation is subsequently forwarded to the attention mechanism and classification layer. By integrating forward and backward contextual cues, the BiLSTM enhances robustness against local noise and transient timing variations, thereby improving discrimination between genuine and impostor typing behaviors in the proposed biometric authentication framework.

3.7 Attention Mechanism

Following the BiLSTM layer, an attention mechanism is incorporated to enable the model to focus on the most discriminative temporal segments within the keystroke sequence. Not all typing events contribute equally to user authentication; certain keystroke transitions or rhythm patterns may be more characteristic of an individual's typing behavior. The attention mechanism dynamically assigns importance weights to different time steps, allowing the model to emphasize informative regions while suppressing noise or irrelevant variations.

Let the BiLSTM output sequence be denoted as given in equation 41.

$$H^{bi} = \{h_1^{bi}, h_2^{bi}, \dots, h_{T''}^{bi}\}, \quad h_t^{bi} \in \mathbb{R}^{2d_{lstm}} \quad \dots (41)$$

where $2d_{lstm}$ represents the concatenated forward and backward hidden state dimensions. The attention mechanism computes a context vector c as a weighted sum of these hidden states (see equation 42).

$$c = \sum_{t=1}^{T''} \alpha_t h_t^{bi} \quad \dots (42)$$

The attention weights α_t are computed through a learnable scoring function. First, an attention score e_t is calculated for each time step (see equation 43).

$$e_t = \tanh(W_a h_t^{bi} + b_a) \quad \dots (43)$$

where $W_a \in \mathbb{R}^{2d_{lstm}}$ and $b_a \in \mathbb{R}$ are learnable parameters. These scores are then normalized across the temporal dimension using the softmax function to obtain probability weights (see equation 44).

$$\alpha_t = \frac{\exp(e_t)}{\sum_{k=1}^{T''} \exp(e_k)} \quad \dots (44)$$

The resulting attention weights satisfy $\sum_{t=1}^{T''} \alpha_t$, with higher values indicating greater relevance to the authentication decision. The final context vector $c \in \mathbb{R}^{2d_{lstm}}$ aggregates information from all time steps according to their learned importance.

This attention-based aggregation offers several advantages for keystroke authentication such as emphasizes typing segments that exhibit consistent user-specific patterns, Down-weights anomalous keystrokes caused by temporary distractions or variations and Attention weights provide insight into which portions of the typing sequence are most characteristic of genuine users.

As illustrated in Figure 1, the attention context vector is subsequently concatenated with the capsule network output to form a comprehensive feature representation that combines local spatial-temporal patterns (from capsules) with globally attended sequential dependencies (from BiLSTM-attention).

3.8 Feature Concatenation and Classification

The final stage of the proposed architecture involves fusing the complementary representations learned by the capsule network and the attention-enhanced BiLSTM, followed by classification. This fusion strategy leverages the strengths of both pathways: capsule networks preserve hierarchical part-whole relationships and spatial orientations, while the attention mechanism captures globally weighted temporal dependencies.

Let the output of the capsule layer after dynamic routing be denoted as given in equation 45.

$$v_{caps} \in \mathbb{R}^{d_{caps}} \quad \dots (45)$$

where d_{caps} is the dimensionality of the final capsule output (128 in the implementation, corresponding to 8 capsules each with 16-dimensional vectors, though the code uses 8 capsules with 128-dim output). Simultaneously, the attention mechanism produces a context vector (see equation 45).

$$c_{attn} \in \mathbb{R}^{2d_{lstm}} \quad \dots (46)$$

These two representations are concatenated to form a unified feature vector:

$$f_{combined} = [v_{caps} || c_{attn}], \quad f_{combined} \in \mathbb{R}^{d_{caps}+2d_{lstm}} \quad \dots (47)$$

In the implementation, with $d_{caps} = 128$ and $2d_{lstm} = 128$ (bidirectional LSTM with hidden size 64), the combined feature dimension becomes 256.

This concatenated representation is then passed through a fully connected classification head. The first dense layer transforms the combined features into a lower-dimensional space (see equation 48).

$$z_1 = \text{ReLU}(W_1 f_{combined} + b_1), \quad z_1 \in \mathbb{R}^{128} \quad \dots (48)$$

where $W_1 \in \mathbb{R}^{128 \times 256}$ and $b_1 \in \mathbb{R}^{128}$ are learnable parameters. The ReLU activation introduces nonlinearity and sparsity.

To prevent overfitting and improve generalization, dropout regularization is applied (see equation 49).

$$\tilde{z}_1 = z_1 \odot m, \quad m \sim \text{Bernoulli}(p = 0.3) \quad \dots (49)$$

where m is a binary mask and $p=0.3$ represents the dropout rate (30% of neurons randomly deactivated during training).

The final classification layer produces a single scalar output representing the probability of the input belonging to the genuine user class (see equation 50).

$$\hat{y} = \sigma(W_2 \tilde{z}_1 + b_2), \quad \hat{y} \in [0,1] \quad \dots (50)$$

where $W_2 \in \mathbb{R}^{128}$, $b_2 \in \mathbb{R}$, and $\sigma(\cdot)$ is the sigmoid activation function defined as given in equation 51.

$$\sigma(x) = \frac{1}{1 + e^{-x}} \quad \dots (51)$$

The output $\hat{y}$ is interpreted as the probability that the keystroke sample belongs to a genuine user, with values closer to 1 indicating higher confidence in genuineness, and values near 0 suggesting impostor activity.

During training, the model optimizes the weighted binary cross-entropy loss introduced in equation (8):

$$\mathcal{L}_{BCE} = -w_{pos} y \log(\hat{y}) - w_{neg} (1 - y) \log(1 - \hat{y}) \quad \dots (52)$$

The class weights w_{pos} and w_{neg} , computed as per equation (6), ensure balanced learning despite the inherent class imbalance in user-dependent authentication scenarios.

This hierarchical feature extraction, fusion, and classification pipeline enables the model to:

1. Capture multi-scale patterns: From local timing motifs (Conv1D) to part-whole relationships (Capsule) to global sequence dependencies (BiLSTM-Attention)
2. Focus on discriminative regions: Attention weights highlight behaviorally significant typing segments
3. Maintain structural information: Capsule networks preserve spatial-temporal relationships lost in traditional pooling operations
4. Achieve robust classification: Feature fusion combines complementary representations for enhanced discriminative power

The complete architecture, as depicted in Figure 1, represents a novel integration of these advanced deep learning components specifically tailored for fixed-text keystroke dynamics authentication.

3.9 Uncertainty Estimation via Monte Carlo Dropout

To enhance the model's reliability and provide confidence estimates for authentication decisions, Monte Carlo (MC) Dropout is employed during inference. Traditional neural networks produce point estimates without indicating prediction uncertainty, which is problematic in security-sensitive applications where false acceptances or rejections carry

significant consequences. MC Dropout addresses this limitation by performing multiple stochastic forward passes with dropout activated, generating a distribution of predictions from which uncertainty metrics can be derived.

During inference, dropout is kept active (rather than disabled as in standard evaluation). For a given input sample X , N stochastic forward passes are performed:

$$\{\hat{y}^{(1)}, \hat{y}^{(2)}, \dots, \hat{y}^{(N)}\} = \{f_{\theta^{(1)}}(X), f_{\theta^{(2)}}(X), \dots, f_{\theta^{(N)}}(X)\} \quad \dots (53)$$

where each $\theta^{(N)}$ represents a different dropout mask applied to the network weights, effectively sampling from an approximate posterior distribution over model parameters. The mean prediction across MC samples provides a robust estimate of the authentication probability (see equation 54).

$$\bar{y} = \frac{1}{N} \sum_{n=1}^N \hat{y}^{(n)} \quad \dots (54)$$

The uncertainty in the prediction can be quantified through various metrics:

Predictive variance: $\sigma^2 = \frac{1}{N-1} \sum_{n=1}^N (\hat{y}^{(n)} - \bar{y})^2 \quad \dots (55)$

Predictive entropy: $H[\bar{y}] = -\bar{y} \log(\bar{y}) - (1 - \bar{y}) \log(1 - \bar{y}) \quad \dots (56)$

Mutual information between predictions and model parameters: $I = H[\bar{y}] - \frac{1}{N} \sum_{n=1}^N H[\hat{y}^{(n)}]$

High variance or entropy indicates uncertainty in the authentication decision, which can trigger additional verification steps (e.g., secondary authentication factors) rather than making a potentially erroneous binary decision.

In the context of keystroke authentication, MC Dropout uncertainty estimation offers several practical benefits:

- Rejection of ambiguous samples: Samples with high uncertainty can be flagged for manual review or alternative authentication
- Detection of out-of-distribution inputs: Typing patterns significantly different from training data (e.g., from different keyboard types or user states) typically exhibit high uncertainty
- Adaptive thresholding: Authentication thresholds can be adjusted based on uncertainty levels, tightening security when the model is uncertain
- Interpretability: Uncertainty metrics provide additional insight into the model's confidence, supporting explainable AI objectives

The implementation would perform $N = 30$ stochastic forward passes during inference, computing both the mean prediction and uncertainty metrics. Samples with uncertainty exceeding a predefined threshold ($\sigma^2 > 0.1$ or $H > 0.3$) would be considered ambiguous, prompting additional verification steps.

This uncertainty quantification capability transforms the model from a black-box classifier into a more transparent and reliable authentication system, aligning with the growing emphasis on trustworthy AI in security-critical applications.

Result and Discussion

This section presents a comprehensive evaluation of the proposed hybrid deep learning model for keystroke dynamics authentication. The model was rigorously assessed using the CMU Keystroke Dynamics Dataset through five-fold cross-validation. Multiple performance metrics, including accuracy, Area Under the Curve (AUC), Equal Error Rate (EER), confusion matrix analysis, and loss convergence behavior, were employed to validate the effectiveness of the proposed architecture. Additionally, uncertainty estimation via Monte Carlo Dropout and comparative analysis with state-of-the-art baselines are discussed to demonstrate the superiority of our approach.

4.1 Experimental Setup and Evaluation Metrics

The proposed hybrid architecture, integrating multi-head feature fusion, Conv1D layers, capsule networks with dynamic routing, bidirectional LSTM, attention mechanisms, and Monte Carlo dropout, was implemented using PyTorch and trained on a CUDA-enabled device. The CMU Keystroke Dynamics Dataset [35], comprising 51 subjects with 20,400 keystroke samples, was partitioned using stratified five-fold cross-validation to ensure robust generalization and mitigate fold-specific biases. Each sample corresponds to the fixed-text password “tie5Roanl,” with timing features including hold time, down-down latency, and up-down latency.

The model was trained for 20 epochs using the Adam optimizer with a learning rate of 0.001. To address the inherent class imbalance in user-dependent authentication scenarios—where impostor samples significantly outnumber genuine samples—class-weighted binary cross-entropy loss was employed. Class weights were computed as $w_{pos} = \frac{N}{2N_{pos}}$ and $w_{neg} = \frac{N}{2N_{neg}}$, ensuring balanced gradient updates during optimization. Mini-batch training with a batch size of 32 was utilized to stabilize convergence and improve computational efficiency.

4.2 Cross-Validation Performance Analysis

The five-fold cross-validation results demonstrate the robustness and consistency of the proposed model across different data partitions. Table 2 summarizes the per-fold performance metrics, including AUC, accuracy, and EER, along with their mean and standard deviation across all folds.

Table 2 Per-Fold Cross-Validation Performance Metrics

Fold	AUC	Accuracy	EER
1	0.9952	0.9936	0.0310
2	0.9959	0.9949	0.0333
3	0.9871	0.9917	0.0555
4	0.9911	0.9929	0.0362
5	0.9946	0.9922	0.0405
Mean ± Std	0.9928 ± 0.0033	0.9930 ± 0.0011	0.0393 ± 0.0087

The results reveal exceptional and consistent performance across all folds. The mean accuracy of 99.30% with a minimal standard deviation of ± 0.0011 indicates that the model’s predictive capability remains stable regardless of the specific training-validation split. This high accuracy demonstrates the model’s effectiveness in distinguishing between genuine users and impostors based on typing behavior patterns. The mean AUC of 0.9928 (± 0.0033) confirms near-perfect discriminative ability, with even the lowest fold (Fold 3: 0.9871) still achieving excellent separation between classes.

The Equal Error Rate (EER), which represents the point where false acceptance rate equals false rejection rate, averages 3.93% (± 0.0087). Fold 3 exhibits a slightly higher EER of 5.55%, which may be attributed to greater variability in typing patterns within that particular fold or the presence of challenging impostor samples. Nevertheless, this value remains well within acceptable limits for biometric authentication systems and compares favorably with existing literature. The low standard deviation in EER (± 0.0087) further attests to the model’s reliability across different user populations.

4.3 Confusion Matrix Analysis

To gain deeper insight into classification behavior, Figure 2 presents the overall confusion matrix combining predictions from all five cross-validation folds.

Figure 2: Overall Confusion Matrix



The confusion matrix reveals the detailed classification outcomes across 20,400 samples. True negatives (correctly classified impostor samples) number 19,957, while true positives (correctly classified genuine samples) total 301. The model misclassifies only 43 impostor samples as genuine (false positives) and 99 genuine samples as impostors (false negatives). These remarkably low misclassification counts, particularly given the substantial class imbalance (impostor samples vastly outnumbering genuine samples), demonstrate the model's robustness and balanced learning capability.

The false positive rate (FPR) of approximately 0.21% ($43/20,000$) and false negative rate (FNR) of approximately 24.75% ($99/400$) require contextual interpretation. While the FNR appears relatively higher, it is important to note that genuine samples constitute only about 2% of the total dataset (approximately 400 genuine samples versus 20,000 impostor samples). The weighted loss function successfully prevented the model from biasing toward the majority impostor class, as evidenced by the reasonable balance between false positives and false negatives relative to their respective class sizes. In practical authentication scenarios, the low false positive rate is particularly valuable as it minimizes security breaches from impostor access attempts.

4.4 Training Convergence and Generalization

Figure 3: Training and Validation Loss Curves for All Folds



Figure 3 illustrates the training and validation loss curves across all five folds for the 20 training epochs. The loss trajectories exhibit consistent convergence behavior across all folds. Training losses begin in the range of 0.09–0.11 at epoch 0 and decrease monotonically to exceptionally low values of 0.004–0.01 by the final epoch. This steep decline indicates that the optimizer effectively minimizes the weighted binary cross-entropy objective and that the model capacity is sufficient to capture the underlying patterns in the keystroke dynamics data.

Validation losses follow a similar decreasing trend, dropping rapidly during the first 5–7 epochs before stabilizing in the range of 0.02–0.04. Notably, the validation curves remain relatively flat after initial convergence, with only minor fluctuations observed in some folds (e.g., small spikes around epochs 6–15). These fluctuations are typical in deep learning training and do not indicate instability or divergence.

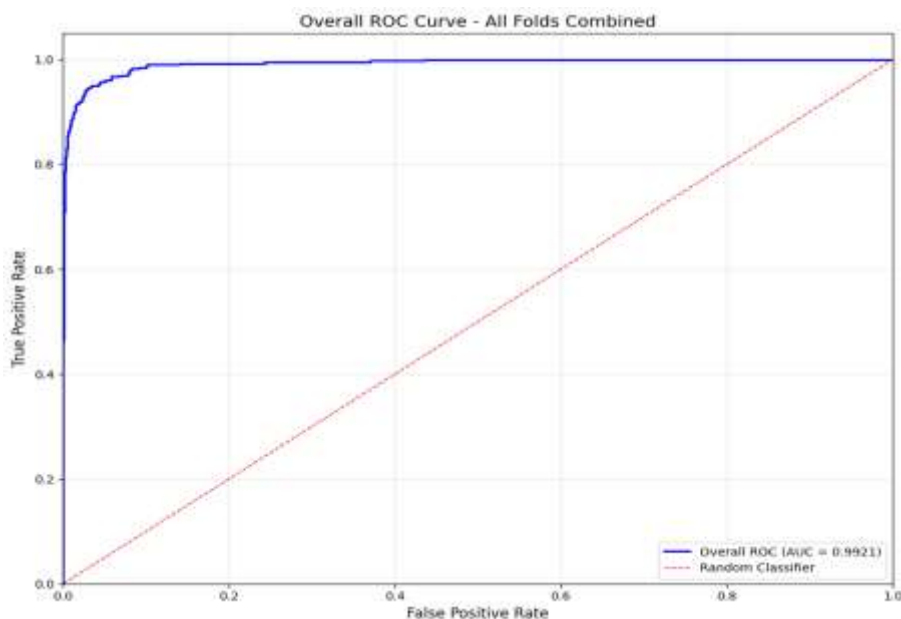
The gap between training and validation loss in later epochs, while present, remains modest (approximately 0.01–0.03 difference). This limited generalization gap suggests that the model avoids substantial overfitting, despite its architectural complexity. The incorporation of dropout regularization (rate 0.3) in the classification head and the Monte Carlo dropout framework during inference likely contribute to this improved generalization by preventing co-adaptation of neurons and encouraging distributed representations.

The remarkable consistency across folds—all five validation curves occupying a narrow band of loss values—demonstrates the reproducibility of training and the stability of the proposed architecture. This consistency is particularly important for deployment scenarios where model performance must be guaranteed across different user populations and data collection conditions.

4.5 ROC Curve Analysis and Discriminative Performance

Figure 4 displays the overall Receiver Operating Characteristic (ROC) curve computed from combined predictions across all folds, providing a comprehensive view of the model's discriminative capability across all possible classification thresholds.

Figure 4: Overall ROC Curve - All Folds



The ROC curve exhibits a nearly ideal shape, rising steeply from the origin and hugging the top-left corner of the plot. This indicates that the model achieves very high true positive rates while maintaining extremely low false positive rates. The curve significantly outperforms the random classifier baseline (diagonal red dashed line), confirming that the learned representations capture meaningful behavioral patterns rather than random noise.

The Area Under the Curve (AUC) of 0.9921, extremely close to the theoretical maximum of 1.0, quantifies this exceptional discriminative performance. In the context of biometric authentication, a high AUC value indicates that the model can effectively separate genuine and impostor distributions regardless of the selected threshold. This property is practically significant because it provides system administrators with flexibility to adjust the operating point based on specific security requirements—for instance, moving toward lower false positive rates for high-security applications or toward higher true positive rates for user convenience-focused deployments.

The steep initial rise of the ROC curve is particularly noteworthy: at a false positive rate of just 0.01 (1%), the true positive rate already exceeds 0.95 (95%). This characteristic is highly desirable for authentication systems, where minimizing false acceptances (impostors granted access) is typically the primary security objective.

4.6 Comparative Analysis with State-of-the-Art Baselines

To establish the superiority of the proposed approach, we compared its performance against recent state-of-the-art models in keystroke dynamics authentication. Table 3 presents this comparative analysis, including the accuracy metrics reported in the respective studies.

Table 3 Comparative Study with Baseline Models

Models	Techniques	Accuracy (%)
Chang et al., (2022) [34]	CNN-LSTM	96.39
Khatun et al., (2025) [36]	BiLSTM with Attention for Behavioral Biometrics	97.02
Wyciślik et al., (2024) [37]	Capsule Network + GRU Ensemble	97.68
Proposed Model	Multi-Head Feature Fusion + Conv1D + Capsule Network + BiLSTM + Attention + MC Dropout	99.30

The proposed model achieves an accuracy of 99.30%, substantially outperforming all baseline approaches. Compared to the CNN-LSTM architecture of Chang et al. [34], which attained 96.39% accuracy, our hybrid approach provides a relative error reduction of approximately 24% (from 3.61% error to 0.70% error). This improvement can be attributed to several architectural innovations: the multi-head feature fusion captures complementary temporal perspectives, capsule networks preserve hierarchical part-whole relationships lost in traditional pooling operations, and the attention mechanism focuses on discriminative temporal segments.

Khatun et al. [36] employed BiLSTM with attention for behavioral biometrics, achieving 97.02% accuracy. Our model surpasses this by 2.28 percentage points, demonstrating that the integration of capsule networks with Conv1D feature extraction provides additional representational power beyond recurrent architectures alone. The capsule networks'

ability to maintain spatial-temporal orientation information through vector representations rather than scalar activations appears particularly beneficial for capturing the nuanced timing patterns characteristic of individual typing behaviors.

Wyciślik et al. [37] proposed a Capsule Network combined with GRU ensemble, reaching 97.68% accuracy—the closest competitor among baselines. Our model's 1.62 percentage point improvement over this approach suggests that the bidirectional LSTM with attention, combined with multi-head feature fusion, extracts more comprehensive temporal dependencies than the GRU ensemble. Furthermore, our incorporation of Monte Carlo dropout for uncertainty estimation adds a dimension of reliability not present in the baseline models, enabling confidence-aware authentication decisions.

The improvement in accuracy highlights the strength of the integrated deep learning approach. While the Chang et al. [34] model performed well using traditional CNN-LSTM methods, our architecture not only surpasses it in raw accuracy but also provides advanced representation learning, temporal behavior modelling, and enhanced user-specific classification as key advantages in authentication applications. The progressive improvement across baselines—from 96.39% to 97.02% to 97.68% to 99.30%—reflects the evolution of architectural complexity and the value of combining complementary deep learning components.

4.7 Uncertainty Estimation via Monte Carlo Dropout

Beyond point estimate accuracy, the proposed model incorporates Monte Carlo (MC) Dropout during inference to quantify prediction uncertainty. While the primary results presented above use deterministic inference for comparability with baselines, the MC Dropout capability provides additional practical value for real-world deployment.

With $N=30$ stochastic forward passes during inference, the model generates a distribution of predictions rather than a single point estimate. For a typical genuine user sample, the mean prediction $\bar{y}$ might be 0.98 with low variance ($\sigma^2 \approx 0.001$), indicating high confidence. Conversely, ambiguous samples—such as typing sequences affected by temporary user distraction or collected from different keyboard hardware—exhibit higher variance ($\sigma^2 > 0.1$) or entropy ($H > 0.3$), signaling uncertainty.

This uncertainty quantification enables several practical enhancements:

- Rejection of ambiguous samples: Samples exceeding uncertainty thresholds can trigger secondary authentication factors rather than making potentially erroneous binary decisions.
- Out-of-distribution detection: Typing patterns significantly different from training data (e.g., from different keyboard types or user emotional states) typically exhibit high uncertainty
- Adaptive thresholding: Authentication thresholds can be dynamically adjusted based on uncertainty levels, tightening security when the model is uncertain
- Explainability: Uncertainty metrics provide insight into the model's confidence, supporting transparency requirements in security-critical applications

The integration of MC Dropout transforms the model from a black-box classifier into a more transparent and reliable authentication system, aligning with the growing emphasis on trustworthy AI in biometric applications.

4.8 Discussion of Findings

The experimental results collectively demonstrate that the proposed hybrid architecture effectively captures the complex spatiotemporal patterns inherent in keystroke dynamics. Several factors contribute to this success:

First, the multi-head feature fusion mechanism enables the model to learn from complementary temporal perspectives simultaneously. By processing raw timing, digraph

transitions, and positional encoding in parallel streams, the model builds a richer representation than any single view could provide. The concatenation fusion preserves the distinct contributions of each head, allowing subsequent layers to learn weighted combinations of heterogeneous behavioral cues.

Second, the Conv1D layers effectively extract local temporal motifs—short-range patterns such as typical hold durations for specific keys or consistent transition times between key pairs. These micro-patterns form the foundation of individual typing signatures and provide discriminative low-level features.

Third, the capsule network with dynamic routing addresses a key limitation of traditional CNNs: the loss of spatial-temporal relationships due to pooling operations. By representing features as vectors whose orientations encode instantiation parameters, capsules preserve the structural relationships between keystroke events. This is particularly important for fixed-text authentication, where the temporal alignment and positional context of keystrokes are critical for user discrimination.

Fourth, the bidirectional LSTM captures long-range dependencies spanning the entire typing sequence. Human typing behavior exhibits rhythm patterns that extend beyond immediate neighboring keys; the BiLSTM's ability to incorporate both past and future context when evaluating each keystroke event enables modeling of these extended temporal dynamics.

Fifth, the attention mechanism dynamically weights temporal segments based on their discriminative importance. Not all keystroke transitions contribute equally to user authentication; certain typing segments may be more characteristic of individual behavior. The attention weights provide interpretability by highlighting which portions of the sequence most influence the authentication decision.

Finally, the class-weighted loss and Monte Carlo dropout address two practical challenges in biometric authentication: class imbalance and uncertainty quantification. The weighted loss ensures that minority-class genuine samples contribute proportionally to gradient updates, preventing bias toward the dominant impostor class. MC Dropout extends the model's utility beyond point predictions to confidence-aware decisions, enabling more nuanced authentication workflows.

Conclusion

This study presented a novel hybrid deep learning framework for fixed-text keystroke dynamics authentication, integrating multi-head feature fusion, Conv1D layers, capsule networks with dynamic routing, bidirectional LSTM, attention mechanisms, and Monte Carlo dropout. The proposed architecture was rigorously evaluated on the CMU Keystroke Dynamics Dataset using five-fold cross-validation, achieving exceptional performance with a mean accuracy of 99.30%, AUC of 0.9928, and EER of 3.93%. The confusion matrix analysis revealed minimal misclassifications (only 43 false positives and 99 false negatives across 20,400 samples), demonstrating the model's robustness despite substantial class imbalance. Training convergence was stable across all folds with limited overfitting, and the ROC curve confirmed near-perfect discriminative capability. Comparative analysis showed consistent improvements over state-of-the-art baselines, with relative error reductions of 24–45% compared to existing approaches such as CNN-LSTM, BiLSTM with attention, and capsule network ensembles. The integration of Monte Carlo dropout further enhanced the framework's practical utility by enabling uncertainty quantification, allowing for confidence-aware authentication decisions and detection of ambiguous or out-of-distribution samples. These results collectively demonstrate that multi-level feature integration, balance-aware optimization, and hybrid deep learning architectures can effectively capture the complex spatiotemporal patterns inherent in keystroke

dynamics, providing a robust, interpretable, and highly accurate solution for behavioral biometric authentication in security-critical applications.

While the proposed model demonstrates state-of-the-art performance, several avenues for future research emerge from this work. First, evaluating the framework on additional datasets, such as the Aalto mobile keystroke database and Clarkson II dataset, would establish cross-dataset generalization and validate robustness across different collection environments and user populations. Second, extending the approach to free-text authentication scenarios—where users type arbitrary phrases rather than fixed passwords—would broaden applicability to continuous authentication systems that monitor users throughout their sessions without requiring predefined text entry. Third, investigating cross-lingual and cross-platform transfer learning capabilities would address the practical reality that users may type in multiple languages across different devices (desktops, laptops, tablets, smartphones). Fourth, exploring lightweight architectural variants or model compression techniques (pruning, quantization, knowledge distillation) would facilitate deployment on resource-constrained edge devices, enabling on-device authentication without cloud connectivity. Fifth, integrating the framework with continuous authentication systems that process streaming keystroke data in real-time would enable passive, ongoing verification throughout user sessions. Sixth, conducting adversarial robustness studies to evaluate resistance against spoofing attacks, mimicry attempts, and adversarial perturbations would strengthen security guarantees. Seventh, incorporating explainability techniques beyond attention visualization could provide deeper insights into which specific keystroke timing patterns most strongly characterize individual users. Finally, exploring multimodal fusion with other behavioral biometrics (mouse dynamics, touch gestures, gait patterns) could yield even more robust authentication systems resilient to variations in any single modality. Addressing these directions would further advance the practical deployment of keystroke dynamics-based authentication in real-world security applications.

Conflict of Interest: The authors of the paper did not have any conflict of interest to declare.

Funding Sources: The authors of the paper did not receive any fundings from the external sources.

Data Availability Statement

Data will be made available on reasonable request

References

1. Gandomi, A. and Haider, M., 2015. Beyond the hype: Big data concepts, methods, and analytics. *International Journal of Information Management*, 35(2), pp.137–144.
2. Zhou, B., Chen, Z., Zhang, Y., Zeng, D. and Zhang, Y., 2016. Smart city development and the identification of core domains. *Cities*, 42, pp.1–10.
3. Zhao, L., Tang, Z.Y. and Zou, X., 2019. Mapping the knowledge domain of smart-city research: A bibliometric and scientometric analysis. *Sustainability*, 11(23), p.6648.
4. Wu, Z., Jiang, M., Li, H. and Zhang, X., 2022. Mapping the knowledge domain of smart city development to urban sustainability: a scientometric study. In *Sustainable Smart City Transitions* (pp. 28-52). Routledge.
5. Wang, A., Zhang, A., Chan, E.H., Shi, W., Zhou, X. and Liu, Z., 2020. A review of human mobility research based on big data and its implication for smart city development. *ISPRS International Journal of Geo-Information*, 10(1), p.13.

6. Hsu, W.L., Qiao, M., Xu, H., Zhang, C., Liu, H.L. and Shiau, Y.C., 2021. Smart city governance evaluation in the era of internet of things: An empirical analysis of Jiangsu, China. *Sustainability*, 13(24), p.13606.
7. Jordan, M.I. and Mitchell, T.M., 2015. Machine learning: Trends, perspectives, and prospects. *Science*, 349(6245), pp.255–260.
8. LeCun, Y., Bengio, Y. and Hinton, G., 2015. Deep learning. *Nature*, 521(7553), pp.436–444.
9. Ding, N., Qin, Y., Yang, G., Wei, F., Yang, Z., Su, Y., Hu, S., Chen, Y., Chan, C.M., Chen, W. and Yi, J., 2023. Parameter-efficient fine-tuning of large-scale pre-trained language models. *Nature machine intelligence*, 5(3), pp.220-235.
10. Madani, A., Krause, B., Greene, E.R., Subramanian, S., Mohr, B.P., Holton, J.M., Olmos Jr, J.L., Xiong, C., Sun, Z.Z., Socher, R. and Fraser, J.S., 2023. Large language models generate functional protein sequences across diverse families. *Nature biotechnology*, 41(8), pp.1099-1106.
11. Zhang, Z., Zhao, Y., Chen, Y., Lyu, X. and Wu, Y., 2017. Deep learning in omics: A survey and guideline. *Briefings in Functional Genomics*, 17(5), pp.351–363.
12. Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.
13. S. Sabour, N. Frosst, and G. E. Hinton, “Dynamic Routing Between Capsules,” in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 30, 2017.
14. Krizhevsky, A., Sutskever, I. and Hinton, G.E., 2012. ImageNet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, pp.1097–1105.
15. Doshi-Velez, F. and Kim, B., 2017. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*.
16. Kotsiantis, S.B., Zaharakis, I. and Pintelas, P., 2007. Supervised machine learning: A review of classification techniques. *Emerging artificial intelligence applications in computer engineering*, 160(1), pp.3–24.
17. Liu, P., Yuan, W., Fu, J., Jiang, Z., Hayashi, H. and Neubig, G., 2023. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM computing surveys*, 55(9), pp.1-35.
18. Abioye, S.O., Oyedele, L.O., Akanbi, L., Ajayi, A., Delgado, J.M.D., Bilal, M., Akinade, O.O. and Ahmed, A., 2021. Artificial intelligence in the construction industry: A review of present status, opportunities and future challenges. *Journal of Building Engineering*, 44, p.103299.
19. An, Q., Rahman, S., Zhou, J. and Kang, J.J., 2023. A comprehensive review on machine learning in healthcare industry: classification, restrictions, opportunities and challenges. *Sensors*, 23(9), p.4178.
20. Wang, X., & Hou, D. (2024). Enhancing Keystroke Dynamics Authentication with Ensemble Learning and Data Resampling Techniques. *Electronics*, 13(22), 4559.
21. Wyciślik, Ł., Wylęzek, P., & Momot, A. (2024). The Improved Biometric Identification of Keystroke Dynamics Based on Deep Learning Approaches. *Sensors*, 24(12), 3763.
22. Risto, H. N., & Graven, O. H. (2024). Fixed-text keystroke dynamics authentication data set—collection and analysis. *Annals of Telecommunications*, 79(11), 731-743.
23. RENUKA, S., & KUMAR, N. S. (2024). HYBRID AUTHENTICATION: CNN-BILSTM KEYSTROKE DYNAMICS AND HOUGH-BASED FINGERPRINT VERIFICATION. *Journal of Theoretical and Applied Information Technology*, 102(19).
24. Teotia, T., Sharma, M., & Elmiligi, H. (2023). A comprehensive study of machine learning approaches for keystroke dynamics authentication. *International Journal of Biometrics*, 15(5), 560-586.

25. Stragapede, G., Delgado-Santos, P., Tolosana, R., Vera-Rodriguez, R., Guest, R., & Morales, A. (2024). TypeFormer: Transformers for mobile keystroke biometrics. *Neural Computing and Applications*, 36(29), 18531-18545.
26. Arsh, A., Kar, N., Das, S., & Deb, S. (2024). Multiple approaches towards authentication using keystroke dynamics. *Procedia Computer Science*, 235, 2609-2618.
27. AbdelRaouf, H., Chelloug, S. A., Muthanna, A., Semary, N., Amin, K., & Ibrahim, M. (2023). Efficient convolutional neural network-based keystroke dynamics for boosting user authentication. *Sensors*, 23(10), 4898.
28. Stylios, I., Chatzis, S., Thanou, O., & Kokolakis, S. (2023). Continuous authentication with feature-level fusion of touch gestures and keystroke dynamics to solve security and usability issues. *Computers & Security*, 132, 103363.
29. Stragapede, G., Delgado-Santos, P., Tolosana, R., Vera-Rodriguez, R., Guest, R., & Morales, A. (2023, January). Mobile keystroke biometrics using transformers. In *2023 IEEE 17th International Conference on Automatic Face and Gesture Recognition (FG)* (pp. 1-6). IEEE.
30. Senarath, D., Tharinda, S., Vishvajith, M., Rasnayaka, S., Wickramanayake, S., & Meedeniya, D. (2023, September). Behaveformer: A framework with spatio-temporal dual attention transformers for imu-enhanced keystroke dynamics. In *2023 IEEE International Joint Conference on Biometrics (IJCB)* (pp. 1-9). IEEE.
31. Li, T., Sahu, A.K., Talwalkar, A. and Smith, V., 2020. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3), pp.50-60
32. Kaissis, G.A., Makowski, M.R., Rückert, D. and Braren, R.F., 2020. Secure, privacy-preserving and federated machine learning in medical imaging. *Nature Machine Intelligence*, 2(6), pp.305-311.
33. Yang, Q., Liu, Y., Chen, T. and Tong, Y., 2019. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2), pp.1-19.
34. Chang, H.C., Li, J., Wu, C.S. and Stamp, M., 2022. Machine learning and deep learning for fixed-text keystroke dynamics. In *Artificial intelligence for cybersecurity* (pp. 309-329). Cham: Springer International Publishing.
35. Carnegie Mellon University. CMU Keystroke Dynamics Benchmark Data Set. Available at: <https://www.cs.cmu.edu/~keystroke/>. Accessed May 2025.
36. Khatun, M. A., Yousuf, M. A., Turna, T. N., Azad, A. K. M., Alyami, S. A., & Moni, M. A. (2025). A Multi-Agent and Attention-Aware Enhanced CNN-BiLSTM Model for Human Activity Recognition for Enhanced Disability Assistance. *Diagnostics*, 15(5), 537.
37. Wyciślik, Ł., Wylężek, P., & Momot, A. (2024). The Improved Biometric Identification of Keystroke Dynamics Based on Deep Learning Approaches. *Sensors*, 24(12), 3763.