IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 4 February 2026, accepted 16 February 2026, date of publication 20 February 2026, date of current version 27 February 2026.

Digital Object Identifier 10.1109/ACCESS.2026.3666649

RESEARCH ARTICLE

KhmerWriterID: Toward Robust Khmer Writer Verification Using Deep Learning: CNN, RNN, and Hybrid Approaches

KIMLONG NGIN[1], DONA VALY[2], SOKKHEY PHAUK[3], VANNARO PIN[4], BAMNANG YIN[5], AND SOKROEURN ANG[6], (Member, IEEE)

[1]Graduate School, Institute of Technology of Cambodia, Phnom Penh 120404, Cambodia
[2]Mechatronics and Information Technology Research Unit, Institute of Technology of Cambodia, Phnom Penh 120404, Cambodia
[3]Department of Applied Mathematics and Statistics, Institute of Technology of Cambodia, Phnom Penh 120404, Cambodia
[4]Faculty of Agriculture, Research Unit, University of Heng Samrin Tboung Khmum, Kandaol Chrum, Tboung Khmum 250503, Cambodia
[5]Institute of Information Technology, University of Heng Samrin Tboung Khmum, Kandaol Chrum, Tboung Khmum 250503, Cambodia
[6]School of AI Computing and Multimedia, Lincoln University College, Petaling Jaya, Selangor 47301, Malaysia

Corresponding author: Kimlong Ngin (nginkimlong@gmail.com)

This work was supported by JICA.

ABSTRACT Writer verification from online handwriting remains a challenging pattern-recognition problem and is particularly underexplored for complex Southeast Asian scripts such as Khmer. This work addresses online, text-independent, word-level Khmer writer verification as a pairwise decision task: given two handwritten word instances, the system determines whether they were written by the same writer. We propose KhmerWriterID, a hybrid Siamese architecture with two input modalities: a grayscale rendering of each word and its pen-trajectory coordinates. In each modality branch, a lightweight CNN extracts spatial features and a bidirectional GRU (BiGRU) encodes stroke dynamics. The resulting modality embeddings are fused into a 128 -dimensional representation, and similarity is computed using scaled cosine similarity. We evaluate KhmerWriterID on a Khmer online handwriting dataset containing both image and coordinate modalities under strict writer-disjoint splits. A Khmer-aware preprocessing pipeline performs sample cleaning, coordinate normalization, scaling to [0, 1], and penstate encoding. Performance is reported using ROC-AUC, EER (equal error rate), and operating-point results at FAR $= 1\%$. KhmerWriterID achieves **99.74**% test accuracy and **99.68**% F1, with validation AUC of **99.92**% and EER of approximately **0.32**%. At FAR $= 1\%$, it attains **99.27**% accuracy and **99.09**% F1 with FRR of approximately **0.07**%. These results suggest that jointly modeling spatial word structure and online stroke dynamics enables robust Khmer writer verification, with potential applications in forensic document analysis, digital authentication, and script-specific handwriting research.

INDEX TERMS KhmerWriterID, writer verification, Khmer handwriting, coordinate sequence, LSTM, custom CNN, hybrid architecture, Siamese neural network (SNN).

I. INTRODUCTION

A. BACKGROUND

Handwriting biometrics captures distinctive motor behaviors that may reflect aspects of a writer's identity and, in some contexts, psychological state [1], [2]. In security and forensic applications, a central challenge is to reliably distinguish individuals based on their handwriting characteristics [3]. This task is challenging because handwriting exhibits substantial variation both between writers and within the same writer across sessions, devices, and writing conditions [4].

In the literature, a clear distinction is typically made between writer identification and writer verification [5]. Writer identification aims to determine which writer in a

The associate editor coordinating the review of this manuscript and approving it for publication was Vincenzo Conti.

candidate set most likely produced a questioned sample (a 1:N problem), whereas writer verification decides whether two given samples were written by the same person or by different people (a 1:1 decision). Another important distinction is between offline and online handwriting. Offline systems operate on static images of ink traces and therefore have limited access to dynamic writing information. Online systems, in contrast, record time-stamped pen trajectories, including (x, y) coordinates and pen-state information, and may additionally capture pressure and speed, enabling richer representations but also requiring robust modeling of temporal dynamics and coping with sensor noise [6], [7].

In this work, we focus on online, text-independent, word-level writer verification for Khmer script. We operationalize verification using a Siamese similarity-learning model that discriminates between same-writer and different-writer pairs of word instances.

Prior work highlights the relevance of handwriting biometrics in security and forensic contexts. For example, Chaski [8] introduced a computational stylometric technique for authorship attribution in digital forensic investigations, and Faundez-Zanuy et al. [6] surveyed handwriting biometrics in esecurity and e-health applications, emphasizing both practical deployments and open research challenges. These studies underline the potential of handwriting biometrics to strengthen digital security and support forensic analyses.

Earlier approaches to writer verification and forensic analysis often relied on handcrafted descriptors and physical characteristics such as writing medium, character size and style, and spacing between characters and words [9]. With the advent of deep learning multilayer neural networks that learn hierarchical representations, writer verification has increasingly shifted toward data-driven adaptive approaches. In particular, trajectory-based methods that analyze sequences of spatial coordinates (x, y) and pen-state indicators (p) have shown strong performance in online settings, and dedicated systems have been developed for major scripts such as English [10], Chinese [11], and Bengali [12], each tailored to the script-specific structure and challenges of the corresponding writing system. However, comparable deep learning-based online writer verification systems for Khmer script remain limited, motivating the present study.

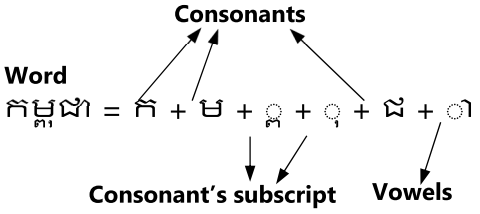

FIGURE 1. A visual representation of Khmer script construction, illustrating how individual characters are formed by combining base consonants with dependent vowels, diacritic symbols, and subscriptconsonants. This multi-layered composition contributes to the script's structural intricacy and presents unique challenges for handwriting analysis.

B. CHALLENGES IN KHMER SCRIPT

Khmer is an abugida writing system of the Mon-Khmer branch of the Austroasiatic language family and is primarily used to write the Khmer language. Its structural richness and large symbol inventory pose substantial challenges for computational handwriting systems [12]. Contemporary Khmer includes 33 basic consonants [13], 17 dependent vowels [14], and around 14 diacritic marks [15] that attach above, below, before, or after the base character. These elements combine into tightly packed consonant-vowel-diacritic clusters with stacked subscripts and complex two-dimensional layouts (Fig. 1), leading to much higher visual and structural complexity than many alphabetic scripts.

For online writer verification, these difficulties are further intensified by substantial intra- and inter-writer variation in stroke order, pen trajectory, and the stylistic realization of the same character cluster. The lack of a large, publicly available online Khmer handwriting dataset and the lack of models specifically designed for Khmer orthographic structure further hinder advancement. Consequently, Khmer writer identification, particularly in the online, text-independent, word-level scenario studied in this work, demands architectures that can simultaneously capture fine-grained spatial patterns and temporal stroke dynamics while remaining robust in low-resource settings.

C. CONTRIBUTIONS

To bridge this gap, we introduce KhmerWriterID, a deep-learning framework for online Khmer handwriting writer verification. Our main contributions are:

- **Hybrid Siamese architecture:** We design a Siamese network that couples a custom CNN for spatial cues with a bidirectional GRU (BiGRU) for temporal dynamics, enabling robust similarity learning between word-level handwriting instances.
- **Curated online Khmer dataset:** We construct a dataset that pairs rasterized word images with online stroke sequences (x, y, p). It includes positive/negative writer pair annotations and standardized preprocessing (normalization, scaling, pen-state encoding).
- **Khmer-aware preprocessing pipeline:** We propose a pipeline tailored to Khmer script, including sequence validation, coordinate normalization and scaling to [0, 1], and explicit pen-state integration to handle multi-stroke characters and inter-device/user variability.
- **Rigorous evaluation protocol:** We evaluate writer verification on word-pair inputs under writer-disjoint train/validation/test splits. Decision thresholds are selected on validation and then frozen for test; we report ROC-AUC, EER, FAR/FRR at the EER operating point, and Accuracy/F1 at a strict operating point of FAR $= 1\%$.
- Empirical insights: KhmerWriterID achieves strong generalization under this protocol and outperforms CNN-only and RNN-only baselines, highlighting the

TABLE 1. List of modern khmer consonants (33 characters)

No.	IPA	ɑ: Series		ɑ: Series	
		C	Sub-C	C	Sub-C
1	k	ក	◌្ក	ក	◌្ក
2	k^h	ខ	◌្ខ	ឃ	◌្ឃ
3	ŋ	ង		ង	◌្ង
4	c	ច	◌្ច	ជ	◌្ជ
5	c^h	ឆ	◌្ឆ	ឈ	◌្ឈ
6	ɲ	ញ		ញ	◌្ញ
7	d	ដ	◌្ដ	ឌ	◌្ឌ
8	t^h	ឋ, ថ	◌្ឋ, ◌្ថ	ឍ, ធ	◌្ឍ, ◌្ធ
9	n	ណ	◌្ណ	ន	◌្ន
10	t	ត	◌្ត	ទ	◌្ទ
11	b	ប	◌្ប	ប៊	
12	p^h	ផ	◌្ផ	ភ	◌្ភ
13	p	ប់		ព	◌្ព
14	m	ម		ម	◌្ម
15	j	យ		យ	◌្យ
16	r	រ		រ	◌្រ
17	l	ឡ	◌្ឡ	ល	◌្ល
18	β	វ		វ	◌្វ
19	s	ស	◌្ស	ស៊	
20	h	ហ	◌្ហ	ហ៊	
21	ʔ	អ	◌្អ	អ៊	

benefit of jointly modeling spatial and temporal cues for Khmer online handwriting.

II. RELATED WORK

Handwriting-based writer verification is a long-standing topic in pattern recognition, tudied under both offline and online settings, with applications in forensic document analysis, identity verification, and digital security. As a behavioral biometric trait, handwriting can encode distinctive motor patterns of an individual and may also correlate with aspects of psychological state [1]. Historically, much of the literature focused on offline writer verification, where models analyze static images of handwritten text. With the widespread adoption of digital pens and touch devices, online handwriting analysis has gained increased attention because it captures temporal and dynamic signals such as stroke order, coordinate trajectories(x, y,pen pressure,and writing speed [16], [17]. These online signals often provide more fine-grained information than offline images alone, which can improve robustness for verification tasks.

The rapid progress of deep learning has further advanced this area by supporting end-to-end representation learning from raw inputs, reducing reliance on handcrafted features. Earlier approaches include models such as Hidden Markov Models (HMMs) and Support Vector Machines (SVMs) while more recent work leverages Recurrent Neural Networks (RNNs), Convolutional Neural Networks (CNNs), and hybrid architectures [18]. A key milestone was introduced by J. Bromley et al. [19], who proposed the Siamese Neural Network(SNN)framework for signature verification. Siamese models have since become widely used in handwriting biometrics because they learn an embedding space in which samples from the same writer are closer than samples from different writers. Instead of predicting a writer class directly, SNNs compute a similarity score (e.g., cosine similarity or Euclidean distance) and apply a threshold to decide whether two samples belong to the same writer.

Recent studies, such as DeepWriterID [20], demonstrate strong performance using deep Siamese architectures that incorporate CNN, recurrent, or hybrid modules to capture temporal and spatial properties of handwriting. These pipelines often include normalization and coordinate scaling as preprocessing steps to standardize inputs and improve training stability [21].

Beyond general frameworks, script-specific writer verification and handwriting recognition systems have also been developed for structurally complex scripts such as Arabic, Bengali, Devanagari, and Chinese, as well as for English [22]. These studies commonly use CNNs, RNNs, or hybrid deep architectures to learn discriminative representations from rendered word images or online stroke trajectories, and they report strong performance under standard biometric protocols (e.g., textindependent writer verification and signature verification). Many of these systems highlight the role of stroke structure, character composition, and script-specific layout in achieving robust verification across varying content and writing conditions.

Compared with these scripts, Khmer introduces additional challenges, including a large consonant inventory, stacked consonant-vowel-diacritic formations, frequent subscripts and combining marks, and complex two-dimensional arrangements within a single word unit (Table 1). Progress is further constrained by the scarcity of publicly available online Khmer handwriting datasets. To the best of our knowledge, prior work has not systematically studied deep online writer verification for Khmer under rigorous, writer-disjoint evaluation protocols. This motivates the present study, which introduces an online Khmer handwriting dataset and a hybrid Siamese architecture designed to capture both temporal stroke dynamics and the spatial structure of Khmer words.

Khmer (/khmaɛ/), also known as Cambodian (ភាសាខ្មែរ /phiəsaː khmaɛ/) and more formally as (ខេមរៈភាសា /khaɛmaːraːphiɜsaː/), is the national language of Cambodia. The language is spoken by the majority of the Cambodian population and by Khmer-speaking diaspora communities in countries such as Vietnam, Thailand, the United States, France, Australia, and Canada [23], [24]. Table 1 summarizes the inventory of the 33 primary consonants used in the modern Khmer script. Despite this broad usage, Khmer has received limited attention in handwriting-based biometric research, reinforcing the need for dedicated study.

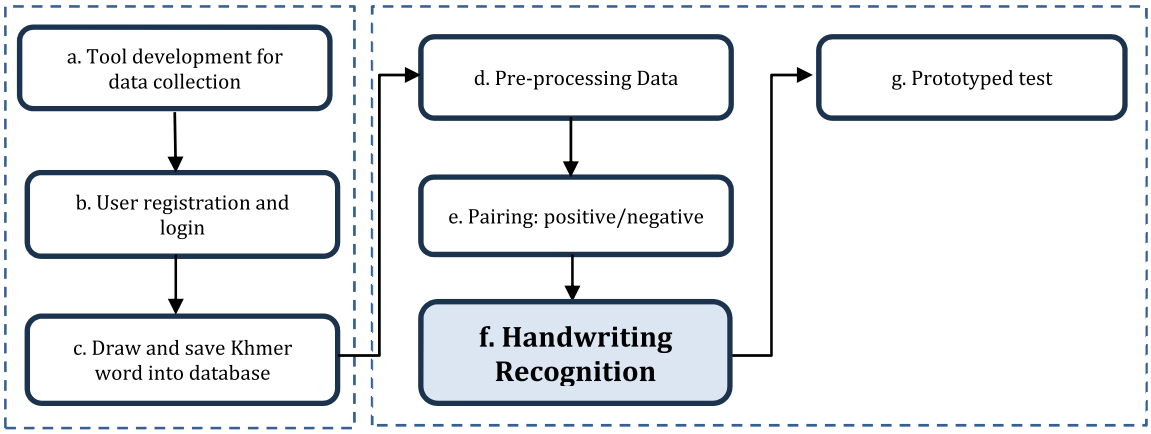

FIGURE 2. Research framework for Khmer handwriting-based writer verification using a Siamese Neural Network (SNN). The framework includes tools for data collection, user registration, and stroke data storage, followed by data preprocessing, sample pairing (positive/negative), and model training. The SNN computes similarity scores where 1 indicates matching writers and -1 indicates different writers using a cosine similarity function. The final stage involves a prototype-based evaluation.

To address this gap, we propose a deep learning approach for online Khmer writer verification using a Siamese architecture that integrates the temporal modeling of stroke sequences with spatial modeling of rendered word images. This motivates the hybrid method introduced in this work.

III. METHODOLOGY

In this study, a comprehensive framework (see Fig. 2) was designed to facilitate both data collection and handwriting preprocessing. The collected handwriting samples, represented as coordinate sequences, were first scaled to a normalized range of $[0, 1]$ to ensure uniformity in the input dimensions. To further enhance consistency across data obtained from various input devices, coordinate normalization was applied to center the handwriting samples spatially. Additionally, pen-state information was embedded within the data to capture essential stroke dynamics, such as pen-down, penup, and end-of-stroke transitions. This enriched representation enables a more accurate characterization of individual writing behaviors.

A. DATASET AND PAIR CONSTRUCTION

We formulate online Khmer writer verification as a binary decision task on pairs of word instances. Each pair is labeled as same writer or different writer and is used to train and evaluate the SNN.

The KhmerWriterID dataset was collected using the Online Tools for Khmer Handwriting (OTKH) platform (Appendix A), which captures online handwriting from multiple Khmer authors. Each sample corresponds to a single handwritten word and includes (i) a grayscale rendered image and (ii) its online stroke trajectory represented as a sequence of (x, y, p) points, along with the associated writer label. In total, the dataset contains 4,878 instances of handwritten words from 298 unique writers, averaging roughly 16 samples per writer.

To train the SNN, we construct pairs of positive and negative handwriting samples from these raw word instances. After pairing all available samples, we initially obtained 241,355 handwriting pairs. We then enforced a writer-disjoint protocol, in which the set of writers is partitioned into nonoverlapping training, validation, and test splits, and pairs are generated only within each split. This guarantees that no writer appears in more than one split and that the evaluation is carried out on completely unseen writers. Under this writer-disjoint setting, the final dataset used in our experiments contains 176,347 pairs, distributed as follows:

- **Training:** 137,269 pairs
- **Validation:** 11,503 pairs
- **Test:** 27,575 pairs

Sample pairing within each split follows the strategy below:

- **Positive pairs:** For each writer, all word samples are paired with other word samples written by the same writer, excluding self-pairings (Fig. 3(A)). These pairs capture intra-writer variability due to differences in lexical content, speed, and writing condition.
- **Negative pairs:** For each sample, we generate 16 random negative pairs by matching it with samples written by different writers within the same split (Fig. 3(B)). This procedure yields a rich set of inter-writer comparisons while keeping the positive/negative ratio suitable for Siamese training.

This structured and writer-disjoint pairing procedure provides a diverse set of positive and negative examples and ensures that the reported results genuinely reflect generalization to unseen Khmer writers, rather than memorization of individuals present in the training set.

1) WRITER-DISJOINT SPLIT & PAIRING

We partition writers into disjoint train/validation/test sets (70/10/20) using a fixed random seed. Pair generation is performed only within each split (no cross-split pairs).

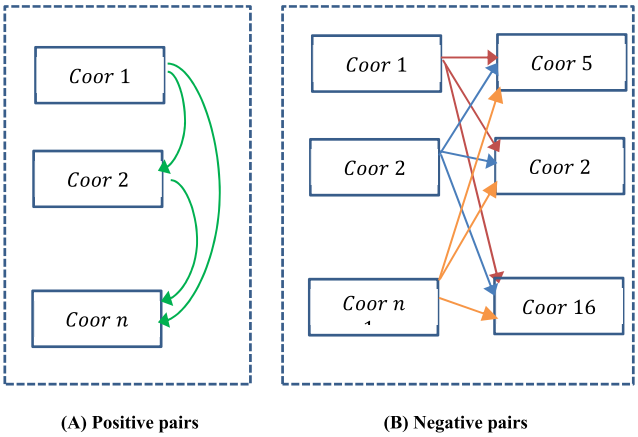

FIGURE 3. Pairing strategy for the KhmerWriterID dataset. Image A (left) illustrates the construction of positive pairs: each handwriting sample from a writer is paired with other samples from the same writer, excluding self-pairings. This captures intra-writer variability in word content and writing conditions. Image B (right) illustrates the construction of negative pairs: for each sample, multiple negative pairs are generated by matching it with samples produced by different writers. This provides diverse interwriter variations for training the Siamese network to discriminate between writers.

- **Positive pairs:** all unordered instance pairs (combinations) from the samewriter within that split.
- **Negative pairs:** sampled only between different writers within the same split, targeting a pos:neg $\approx$ 1:1 ratio (unless noted)

2) VALIDATION-FIXED THRESHOLDS AND TEST EVALUATION

On the validation split, we sweep similarity thresholds to locate the equal-error threshold τ_{EER} where FAR(false acceptance rate) and FRR(false rejection rate) intersect (with linear interpolation). We also set $\tau_{FAR} = \alpha$ to achieve a target impostor acceptance (e.g., $\alpha = 1\%$) by taking the $(1-\alpha)$ - quantile of the validation impostor-score distribution. These thresholds are then frozen and applied to the test split to report:

- At τ_{EER}: ERR and FAR/FRR@EER;
- At $\tau_{\text{FAR}} = 1\%$: Acc/F1/FRR at FAR $= 1\%$.

B. INCORPORATING PEN-STATE INTO COORDINATE DATA

To enhance the handwriting data with temporal stroke dynamics, a pen-state feature was introduced alongside the (x, y) coordinates, resulting in triplet representations of the form $(\text{x}_\text{t}, \text{y}_\text{t}, \text{p}_\text{t})$, where t denotes the time step. The pen-state p_t encodes critical writing actions:

$$S = \{(x_t, y_t)\}_{t=1}^{T} \, x_t, y_t \in \mathbb{R} \tag{1}$$

The pen-state variable p_t encodes writing activity, defined as: $p_t \in \{0, 1, 2\}$

- $p_t = 0$ represent pen-down (actively writing)
- $p_t = 1$ represent Pen-up (pen lifted)
- $p_t = 2$ represent end of sequence

Each data point is then extended to:

$$s_t = (x_t, y_t, p_t) \in \mathbb{R}^2 \times \{0, 1, 2\} \tag{2}$$

The full sequence is represented as:

$$S' = \{(x_t, y_t p_t)\}_{t=1}^{T} \tag{3}$$

where,

T is the total number of points in the sequence. This structured representation enables the model to capture both spatial patterns and writing behaviors, which are essential for tasks of writer verification.

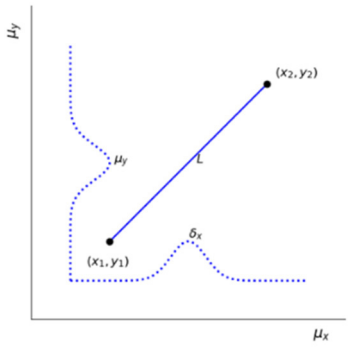

FIGURE 4. Illustration of the coordinate normalization process applied to handwriting data. This step standardizes the (x, y) values across samples to ensure consistency in scale and spatial alignment, regardless of input device or writing style.

C. FEATURE SCALING

To address variability in stroke size across handwriting samples, all coordinate values were normalized to the [0, 1] range using min-max scaling. This normalization helps improve training stability and model convergence by minimizing the risk of large gradient updates and ensuring compatibility with commonly used activation functions. The min-max normalization is defined as follows:

$$x_{\text{scaled}} = \frac{x - x_{\text{min}}}{x_{\text{max}} - x_{\text{min}}} \tag{4}$$

$$y_{\text{scaled}} = \frac{y - y_{\text{min}}}{y_{\text{max}} - y_{\text{min}}} \tag{5}$$

where,

- x, y represent the raw coordinate values of each point in the stroke sequence.
- x_{min}, x_{max} are the minimum and maximum x -values in a given handwriting sample.
- y_{min}, y_{max} are the corresponding minimum and maximum y-values.
- x_{scaled}, y_{scaled} are the resulting normalized coordinates within the [0, 1] range.

To maintain the structural integrity of stroke boundaries during preprocessing, a sentinel value of -1 was inserted as a marker to denote breaks between individual strokes. This strategy enables the model to distinguish between continuous and segmented strokes without disrupting the normalized coordinate range.

D. COORDINATE NORMALIZATION

Another source of variability arises from differences in the scale and absolute values of the coordinate data, which may result from diverse input devices or individual handwriting styles. To ensure consistency across all samples, it is necessary to normalize the (x, y) coordinates to a fixed interval. As illustrated in Fig. 4, a straight line L connects two points (x_1, y_1) and (x_2, y_2), serving as an example of spatial alignment and scale representation.

The projections of this line onto the x -axis and y -axis are defined as:

$$p_x(\mathrm{L}) = \int_{\mathrm{L}} \mathrm{x}\, \mathrm{d}\, \mathrm{L} = \frac{1}{2}\, \mathrm{len(L)}\, (\mathrm{x_1} + \mathrm{x_2})\,, \tag{6}$$

$$p_y(\mathrm{L}) = \int_{\mathrm{L}} \mathrm{y}\, \mathrm{d}\, \mathrm{L} = \frac{1}{2}\, \mathrm{len(L)}\, (\mathrm{y_1} + \mathrm{y_2})\,, \tag{7}$$

where $l\mathrm{en(L)} = \sqrt{(\mathrm{x_2} - \mathrm{x_1})^2 + (\mathrm{y_2} - \mathrm{y_1})^2}$ present the L length. With this information, we could calculate the mean values by projecting all lines onto the x - and y - axes:

$$\mu_{\mathrm{x}} = \frac{\Sigma_{\mathrm{L}\in\Omega} \mathrm{p_x(\,L)}}{\Sigma_{\mathrm{L}\in\Omega}\, \mathrm{len(\,L)}}, \mu_{\mathrm{y}} = \frac{\Sigma_{\mathrm{L}\in\Omega} \mathrm{p_y(\,L)}}{\Sigma_{\mathrm{L}\in\Omega}\, \mathrm{len(\,L)}} \tag{8}$$

where Ω represents the set of all straight lines connecting successive points within the same stroke. We then computed the deviation from the mean of the x - and y -axis projections to assess stroke variability:

$$d_x(\mathrm{L}) = \int_{\mathrm{L}} (\mathrm{x} - \mu_{\mathrm{x}})^2\, \mathrm{dL} = \frac{1}{3}\, \mathrm{len}(L) \Big[(\mathrm{x_2} - \mu_{\mathrm{x}})^2 + (\mathrm{x_1} - \mu_{\mathrm{x}})^2 + (\mathrm{x_1} - \mu_{\mathrm{x}})\, (\mathrm{x_2} - \mu_{\mathrm{x}})\Big] \tag{9}$$

The standard deviation along the x -axis can be estimated using the following formula:

$$\delta_{\mathrm{X}} = \sqrt{\frac{\Sigma_{\mathrm{L}\in\Omega} \mathrm{d_x(L)}}{\Sigma_{\mathrm{L}\in\Omega}\, \mathrm{len(L)}}} \tag{10}$$

With all the information of μ_x, μ_y and δ_x estimated from one character, we can now normalize the coordinates by:

$$\mathrm{x_{new}} = \frac{(\mathrm{x} - \mu_{\mathrm{x}})}{\delta_{\mathrm{x}}}, \mathrm{y_{new}} = \frac{\big(\mathrm{y} - \mu_{\mathrm{y}}\big)}{\delta_{\mathrm{x}}} \tag{11}$$

This normalization is applied globally to all points within the character. Notably, the standard deviation along the y -axis is not estimated; instead, the y -coordinates are also normalized using δ_x. This approach is adopted to retain the original height-to-width ratio of the character and preserve the directional integrity of each stroke. Following normalization, each character is mapped onto a standardized Cartesian coordinate system, while its overall shape and structural proportions are maintained.

E. RENDERING COORDINATE SEQUENCE INTO IMAGES

To prepare pen-based stroke sequences for CNN-based classification, each handwriting sample is transformed into a grayscale image (Algorithm 1). Consider a stroke sequence $S = \{\mathrm{s_1}, \mathrm{s_2}, \ldots, \mathrm{s_n}\}$ where each sub stroke $\mathrm{s_i} = \{(\mathrm{x_i}, \mathrm{y_i}, \mathrm{p_i})\}_{\mathrm{j}=1}^{\mathrm{m_i}}$ consists of a series of coordinate points paired with penstate values. The pen-state $\mathrm{p_i} \in \{0, 1, 2\}$ indicates whether the pen is in contact with the surface (pendown) or lifted (penup).

The rendering process outputs an image with a target resolution $\in \mathbb{Z}_+$, typically set to 128×128 pixels. To enhance stroke clarity and apply anti-aliasing, the rendering is initially performed at a higher resolution determined by an integer upscale factor upscale_factor $\in \mathbb{Z}_+$, then resized to the target resolution. Additional parameters include line_width $\in \mathbb{Z}_+$, which defines the stroke thickness, and margin $\in \mathbb{Z}_+$, which allocates padding around the rendered content to prevent edge clipping. This rendering pipeline is implemented using the Python Imaging Library (PIL), which supports precise drawing operations and high-quality resampling methods such as Lanczos interpolation for final image resizing.

Algorithm 1 Rendering Algorithm for Converting Stroke Sequences to Grayscale Image.

Step	Operation
Input	Input $S = \{s_1, s_2, \ldots, s_n\}$, image size, upscale factor u
Output	Grayscale image $I \in [0, 1]^{size \times size}$
1	$h \leftarrow size \times u \mid m \leftarrow margin \times u, t \leftarrow line_{width} \times u$
2	$P \leftarrow \{(x, y) \mid x \geq 0, y \geq 0, (x, y, p) \in s, s \in S\}$
3	If $P \leftarrow= \emptyset$, *return white image* $I \in \mathbb{R}^{size \times size}, I(i, j) = 1$
4	$min_x, max_x \leftarrow \frac{min}{max_{(x,y)}} \in pX$, same for y
5	$\Delta_x \leftarrow max_x - min_x + \epsilon, \Delta_y \leftarrow max_y - min_y + \epsilon$
6	$s \leftarrow h - 2m$
7	Initialize white canvas $H \in \mathbb{R}^{h \times h}$, with pixel values 255
8	**FOR EACH** substroke $s \in S$:
9	Set $prev \leftarrow None$
10	**FOR EACH** *point* $(x, y, p) \in s$:
11	IF $x < 0 vy < 0$: *set prev* $\leftarrow$ *None*; *continue*
12	Normalize: $px \leftarrow m + \frac{(x - mn_x)}{\Delta_x}.S$
13	$py \leftarrow m + \frac{(x - \min_y)}{\Delta_y}.S$
14	**IF** $p = 0 \Lambda$ *prev* $\neq$ None: draw line from prev to ($px, p\, y$)
15	*Set prev* $\leftarrow$ None
16	**IF** $p = 1$: set *prev* $\leftarrow$ None
17	**END FOR**
18	Resize $H \leftarrow I \in \mathbb{R}^{\mathrm{size} \times size}$ using Lanczos resampling
19	Feature scale $I \leftarrow I/255.0$
20	**RETURN** I
21	**END FOR**

IV. MODEL ARCHITECTURE

The KhmerWriterID hybrid Siamese architecture is designed to process dual input streams grayscale handwriting images and pen-based stroke coordinates using two parallel, identical subnetworks (Fig. 5). Each branch independently extracts modality-specific features from the image and stroke-sequence inputs. The resulting representations are then fused and projected into a shared embedding space to evaluate writer similarity. The architecture comprises the following key components:

- **Conv2D layers (image stream):** Capture local spatial patterns from input images. For example, a Conv2D layer with parameters (1 → 32,"kernel" = 5, "padding" = 2) maps an input tensor of shape [B, 1, 64, 64] to [B, 32, 64, 64].
- **Max pooling (MaxPool2D):** Reduces spatial resolution while preserving salient structures. A 2 × 2 MaxPool maps [B, 32, 64, 64] to [B, 32, 32, 32], improving computational efficiency and robustness.
- **ReLU activation:** Introduces nonlinearity after each convolution, enabling the model to learn complex handwriting patterns.
- **Deep CNN feature extractor + FC projection:** After three Conv2D-ReLU-MaxPool blocks, the feature map has shape [8, B,128,bib:8]. It is flattened to 8192 dimensions and passed through an MLP projection head Linear (8192 → 512 → 128) with ReLU and BatchNorm to produce a 128-D image embedding.
- **Batch normalization (BatchNorm1d):** Normalizes intermediate activations (e.g., BatchNorm1d(512)) to stabilize optimization and speed convergence.
- **Dropout:** Applies dropout with $p = 0.3$ to reduce overfitting during training.
- **Bidirectional GRU (BiGRU, coordinate stream):** Processes the sequential input (x, y, p). The BiGRU with input size 3 and hidden size 64 per direction produces an output of shape $[B, T, 128]$. The last time step (or an equivalent aggregation) yields a fixed-length vector $[B, 128]$ summarizing pen-trajectory dynamics.
- **Coordinate projection (MLP):** The BiGRU summary is passed through a small projection head Linear (128 → 256 → 128) with ReLU, BatchNorm, and Dropout to produce a 128-D coordinate embedding capturing stroke dynamics.
- **Cross-modal fusion (concatenation):** The image and coordinate embeddings are fused by a single concatenation step: $f_{\text{fusion}} = [f_{\text{img}}; f_{\text{coor}}] \in \mathbb{R}^{256}$, yielding a fused feature vector of shape [B, 256].
- **Fusion head (dense projection):** A final FC projection maps the fused 256-D vector to a 128-D writer embedding optimized for discriminative comparison.
- **L2 normalization and learnable scaling:** The final embedding is L2-normalized and multiplied by a learnable scale factor (initialized to 10.0) to stabilize cosine-similarity training.

The Siamese framework comprises two weight-sharing subnetworks that independently process the two samples in a pair. Because both branches share parameters, they produce comparable embeddings that can be directly contrasted via the similarity function. To identify an effective design for Khmer handwriting, we evaluated multiple encoder variants (CNN-only, GRU-based sequence-only, and hybrid CNN+BiGRU), each targeting complementary spatial and temporal characteristics of Khmer script [25].

To effectively adapt the Siamese architecture to the unique structural complexity of Khmer script, we implemented task-specific preprocessing techniques. This included stroke normalization, feature scaling, and noise reduction to improve the clarity of diacritics and minimize background artifacts common challenges in Khmer handwriting. Although the model operates on sequential coordinate data, it retains key aspects of online handwriting dynamics captured via the data collection tool, including pen trajectory, stroke order, and pen-state transitions. This ensures that temporal characteristics, such as writing direction and stroke flow, are preserved during preprocessing, enabling the model to fully leverage dynamic handwriting patterns within a coordinate-based framework.

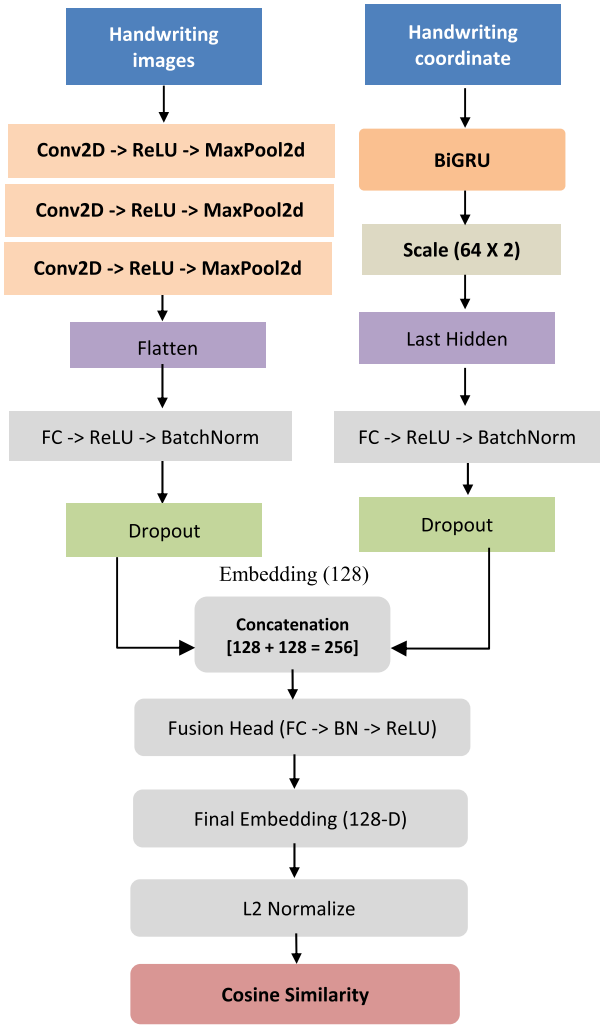

FIGURE 5. The KhmerWriterID framework uses a hybrid Siamese architecture that processes two complementary input modalities: grayscale handwriting images and pen-based stroke coordinate sequences. Each side of the Siamese network consists of identical, weight-sharing subnetworks. The CNN branch extracts spatial features from the rendered grayscale image, while the BiGRU branch encodes temporal writing dynamics from the coordinate sequence (x, y, pen-state) The resulting 128-dimensional embeddings from both branches are concatenated and passed through a fusion head composed of fully connected layers to produce a unified writer representation. Cosine similarity between the paired embeddings is then computed to measure writing similarity for the writer verification task.

The similarity between two handwriting embeddings was measured by a cosine similarity function, which is widely adopted in Siamese architectures for measuring the angular closeness in the feature space. The cosine similarity score S between two embedding vectors $f\left(x^i\right)$ and $f\left(x^j\right)$ is defined as:

$$\mathbf{S} = \cos\left(\mathrm{f}\left(\mathrm{x}^{\mathrm{i}}\right), \mathrm{f}\left(\mathrm{x}^{\mathrm{j}}\right)\right) = \frac{\mathrm{f}\left(\mathrm{x}^{\mathrm{i}}\right) \cdot \mathrm{f}\left(\mathrm{x}^{\mathrm{j}}\right)}{\left\|\mathrm{f}\left(\mathrm{x}^{\mathrm{i}}\right)\right\| \cdot \left\|\mathrm{f}\left(\mathrm{x}^{\mathrm{j}}\right)\right\|} \tag{12}$$

where,

- $f\left(x^i\right) \in \mathbb{R}^n$ and $f\left(x^j\right) \in \mathbb{R}^n$ are the learned embedding vectors for the two input sequences
- $\|.\|$ denotes the L2-norm (Euclidean norm)
- n is the dimensionality of the embedding space.

The resulting similarity score $S \in [-1, 1]$ indicates how aligned the two embeddings are: $S = 1$: vectors are perfectly aligned (maximum similarity)

- $S = 0$: vectors are orthogonal (unrelated)
- $S = -1$: vectors are diametrically opposed (maximum dissimilarity, rarely used due to L2

For each pair of word instances $\left(\mathrm{x_i}, \mathrm{x_j}\right)$, the two branches of the Siamese network produce 128-D embeddings $f\left(x_i\right)$ and $f\left(x_j\right)$. Their cosine similarity S is used as the verification score. Within the biometric evaluation protocol of Section III, a single global decision threshold τ is selected on the validation set (either at the equal-error-rate operating point, τ_{EER} or at the FAR $= 1\%$ operating point $\tau_{\mathrm{FAR}} = 1\%$). At test time, this threshold τ is held fixed: a pair is classified as "same writer" if $\mathrm{S} \geq \tau$ and classified as "different writer" otherwise. The resulting similarity scores are then used to compute the ROC curves, AUC, EER, FAR, FRR, accuracy, and F1-score.

V. RESULTS

A. TRAINING ENVIRONMENT SETUP

All procedures for data collection, preprocessing, experimentation, and evaluation were performed using the Python programming language. The deep learning models were developed and trained with the PyTorch framework. To maintain consistency and reproducibility across experiments, the following hyperparameters were employed:

- Machine: NVIDIA GeForce RTX 4090 GPU
- Learning rate: 0.0001
- Optimizer: AdamW (lr 1×10^{-4}, weight decay 1×10^{-5}).
- Batch size: 32.
- Random seeds: {11, 33, 55}. When aggregating, report the mean std across seeds.
- Protocol: writer-disjoint splits; thresholds selected on validation and frozen for test.

All models were trained, validated, and tested using a contrastive learning framework, with cosine similarity employed as the primary metric to measuring distance between paired embeddings. Model performance was evaluated based on accuracy and F1-score across the training, validation, and testing datasets to comprehensively assess the generalization capabilities of the different Siamese neural network (SNN) architectures.

TABLE 2. Training schedule and configurations for all total epochs of the siamese model, best validation epoch, and backbone / RNN stack.

Model	Epochs	Best epoch (validation)	Backbone/ RNN Stack
DeepWriterID	30	29	DeepWriterID-CNN (image-only)
AlexNet	50	49	AlexNet (image-only)
ResNet	20	19	ResNet-18 (image-only)
VGGNet	20	20	VGG-16-BN (image-only)
LSTM	50	50	LSTM 1×64 (seg-only)
GRU	50	50	GRU 2×128 (seg-only)
BiGRU	50	49	BiGRU 2×128 (seg-only)
BiLSTM	50	46	BiLSTM 1×64 (seg-only)
GRU + LeNet	50	50	LeNet + GRU 1×64 (fusion)
LSTM + LeNet	50	49	LeNet + LSTM 1×64 (fusion)
LSTM + AlexNet	50	46	AlexNet + LSTM 1×64 (fusion)
BiLSTM + Simple CNN	50	48	Simple CNN + BiLSTM 1×64) (fusion)
KhmerWriterID	50	48	CNN + BiGRU (1 layer, hidden 64 per direction → 128-D)

B. EVALUATION METRICS AND PROTOCOL

We evaluated all systems under a writer verification (1:1) protocol. Each input is a pair of handwriting samples (x_i, x_j) labeled as genuine (same writer) or impostor (different writers). The Siamese network outputs a cosine-similarity score $\mathrm{s} \in [-1, 1]$; a pair is accepted as same-writer if $\mathrm{s} \geq \tau$ and rejected otherwise. All splits are writer-disjoint (train/validation/test) to avoid identity leakage. Classification outcomes as a function of the decision threshold τ:

- TP(τ): genuine (same-writer) pairs correctly accepted
- FN(τ): genuine pairs incorrectly rejected
- TN(τ): impostor (different-writer) pairs correctly rejected
- FP(τ): impostor pairs incorrectly accepted

Siamese embeddings:

$$\mathbf{z_a} = \mathrm{f}_\theta\left(\mathrm{x_a}\right), \mathbf{z_b} = \mathrm{f}_\theta\left(\mathrm{x_b}\right) \tag{13}$$

Cosine similarity

$$\mathbf{s} = \cos\left(z_a, z_b\right) = \frac{z_a^T z_b}{|z_a\|_2 \mid z_b\|_2} \tag{14}$$

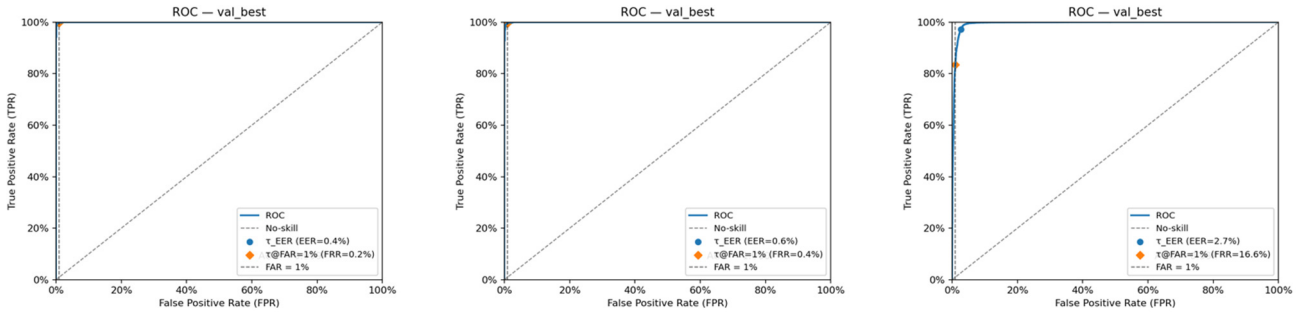

FIGURE 6. Validation ROC curves (CNN-only). From left to right: VGG-16-BN, ResNet, and DeepWriterID. Markers denote the validation-selected operating points at τEER and at τ_{FAR = 1%} - with the vertical dashed line indicating FAR = 1% and the diagonal showing the no-skill baseline. Reported Values: DeepWriterID EER ≈ 2.7%, FRR @1% FAR ≈ 16.6%, VGG-16 EER ≈ 0.4%, FRR @1% FAR ≈ 0.2%; ResNet EER ≈ 0.6% FAR ≈ 0.2%. Thresholds are chosen on validation and then fixed for test metrics (Table 3).

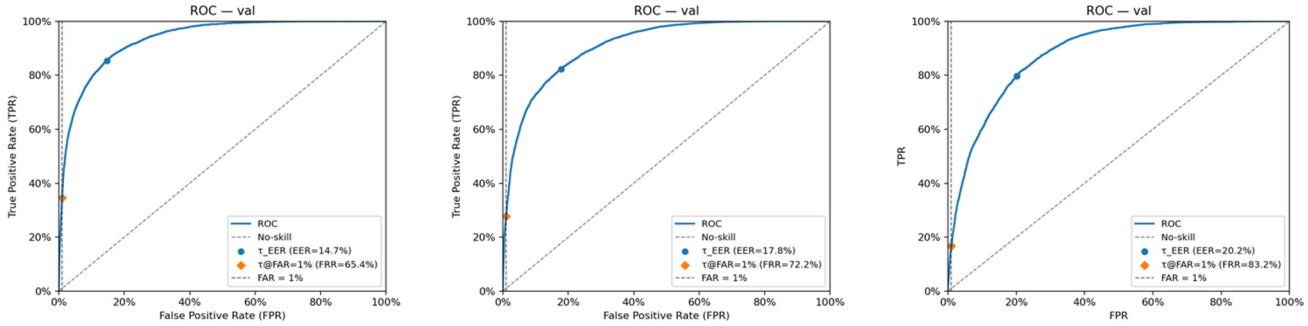

FIGURE 7. Validation ROC curves (RNN-only). From left to right: BiGRU, LSTM, and BiLSTM. Markers show the validation-selected operating points (blue) at τEER and at T_{FAR = 1%}. Approximate validation stats read from the plots: BiGRU EER ≈ 14.7%, FRR@1%FAR ≈ 65.4%; LSTM EER =17.8%,FRR@1%FAR ≈ 72.2%; BiLSTM EER ≈ 20.2% FRR@1%FAR ≈ 83.2%. Thresholds are chosen on validation and then held fixed for the test metrics reported in Table 4.

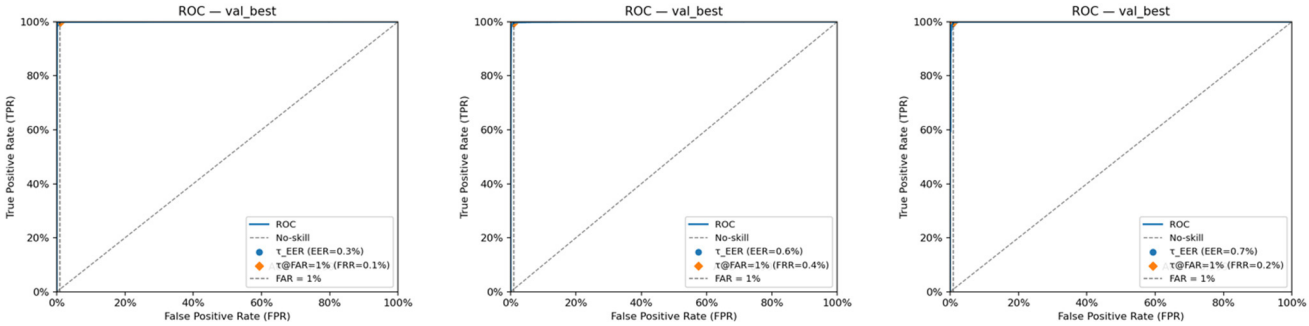

FIGURE 8. Validation ROC (best epoch) for hybrid Siamese models. From left to right: KhmerWriterID (CNN+BiGRU), LSTM+ SimpleCNN, and LSTM+AlexNet. Markers show the validation-selected operating points at τEER (EER ≈ 0.3%, 0.6%, 0.7%) and at τ_{FAR = 1%}(FRR@1 ≈ 0.4%, 0.2%).

Decision rule at threshold τ:

$$\hat{y}(\tau) = \begin{cases} 1, & s \geq \tau \\ 0, & s < \tau \end{cases} \tag{15}$$

Total numbers of genuine and impostor pairs

$$\mathrm{N_{pos}} = \mathrm{TP}(\tau) + \mathrm{FN}(\tau), \mathrm{N_{neg}} = \mathrm{TN}(\tau) + \mathrm{FP}(\tau) \tag{16}$$

Error rates

$$\mathrm{FAR}(\tau) = \frac{\mathrm{FP}(\tau)}{\mathrm{N_{neg}}}, \mathrm{FRR}(\tau) = \frac{\mathrm{FN}(\tau)}{\mathrm{N_{pos}}} \tag{17}$$

On the validation set, we sweep over thresholds $\boldsymbol{\tau}$ to obtain equal error rate $\boldsymbol{\tau_{\mathrm{EER}}}$ threshold and a strict $\mathbf{FAR} = \mathbf{1}\%$ operating point.

$$\tau_{\mathrm{EER}} = \arg\min_{\tau} |\,\mathrm{FAR\ val}(\tau) - \mathrm{FRR\ val}(\tau)| \tag{18}$$

On the validation set, we also choose a threshold that enforces approximately 1% FAR:

$$\tau_{\mathrm{FAR}} = 1\% = \arg\min_{\tau} |\,\mathrm{FAR\ val}(\tau) - 0.01| \approx \mathrm{Q0.99}\,(\{s_i \mid (s_i, y_i) \in s_{val}, y_i = 0\}) \tag{19}$$

TABLE 3. CNN-only siamese verification on khmer handwriting (DeepWriteriD, AlexNet, VGG-16-BN, ResNet-18). Validation thresholds τ_{EER} and $\tau_{FAR=1\%}$ are chosen on validation and then frozen for test. We report FAR/FRR at τ_{EER} (vAL) and ACC/F 1/FRR @ 1% FAR at $\tau_{FAR=1\%}$ (VAL). τ denotes the decision threshold on the similarity score.

Model	Val AUC (%)	Val EER (%)	Test AUC (%)	FAR@ τ_{EER} (Val) (%)	FRR@ τ_{EER} (Val) (%)	Acc@FAR=1% (%)	F1@FAR=1% (%)	FRR@FAR= 1% (%)
DeepWriterID	99.41	2.73	99.41	2.73	2.73	92.75	90.22	16.59
ResNet	99.95	0.65	99.96	0.66	0.66	99.24	99.05	0.27
VGGNet	99.95	0.41	99.95	0.48	0.31	99.38	99.22	0.14

TABLE 4. RNN-only Siamese verification on sequence-only inputs (LSTM, GRU, BiGRU, BiLSTM) under the same validation selected operating points.

Model	Val AUC (%)	Val EER (%)	Test AUC (%)	FAR@ τ_{EER} (Val) (%)	FRR@ τ_{EER} (Val) (%)	Acc@FAR=1% (%)	F1@FAR=1% (%)	FRR@FAR=1 % (%)
LSTM	91.09	17.76	90.67	17.98	17.59	70.58	42.59	72.48
GRU	89.03	19.94	88.47	20.61	20.98	68.84	37.29	76.63
BiGRU	93.40	14.68	93.06	15.01	14.73	72.89	49.29	66.77
BiLSTM	89.01	19.17	88.79	19.57	19.39	63.03	27.03	83.31

TABLE 5. Hybrid CNN+RNN siamese verification (GRU+LeNet, LSTM+LeNet, LSTM+AlexNet, BiLSTM+SimpleCNN, KhmerWriteriD) with metrics computed at τ_{EER} (VAL) and $\tau_{FAR=1\%}$ (VAL).

Model	Val AUC (%)	Val EER (%)	Test AUC (%)	FAR@ τ_{EER} (Val) (%)	FRR@ τ_{EER} (Val) (%)	Acc@FAR=1% (%)	F1@FAR=1% (%)	FRR@FAR= 1% (%)
GRU + LeNet	99.53	1.81	99.52	1.90	1.79	97.72	97.08	4.05
LSTM + LeNet	99.52	1.77	99.53	1.97	1.53	97.79	97.18	3.87
LSTM + AlexNet	99.86	0.69	99.88	0.78	0.68	99.28	99.09	0.27
BiLSTM + Simple CNN	99.92	0.59	99.93	0.59	0.47	99.31	99.13	0.34
KhmerWriterID	**99.92**	**0.32**	**99.92**	**0.30**	**0.19**	**99.27**	**99.09**	**0.07**

where $s_{val} = \{(s_i, s_j)\}$ is the validation set; s_i = Similar score; $y_i \in \{0, 1\}$ (1= genuine, 0 = imposter); $Q0.99$ = empirical 99^{th} percentile (right-continuous interpolation).

Precision, Recall, Accuracy, F1(used atτ_{EER} and τ_{FAR}).

(20)

$$\text{Acc}(\tau) = \frac{\text{TP}(\tau) + \text{TN}(\tau)}{\text{N}_{pos} + \text{N}_{neg}} \tag{20a}$$

$$\text{Precision}(\tau) = \frac{\text{TP}(\tau)}{(\text{TP}(\tau) + \text{FP}(\tau))} \tag{20b}$$

$$\text{Recall}(\tau) = \frac{\text{TP}(\tau)}{\text{TP}(\tau) + \text{FN}(\tau)} \tag{20c}$$

$$F1(\tau) = \frac{2 \times \text{ Precision } (\tau) \times \text{ Recall } (\tau)}{\text{Precision } (\tau) + \text{ Recall } (\tau)} \tag{20d}$$

Figures mark the validation-selected thresholds: the equal-error threshold τ_{EER} and the τ_{FAR} = 1% line. The tables report all the metrics at these thresholds. Unless otherwise noted, the results are averaged over the specified random seeds and reported as mean SD ± (standard deviation); the best epoch is selected by the lowest validation EER.

C. RESULT AND COMPARISON

All results below follow the writer-verification protocol with writer-disjoint splits. Thresholds τ_{EER} and $\tau_{FAR=1\%}$ are selected once on validation and then frozen for test. Tables 3-5 report ROCAUC, validation EER, the corresponding thresholds, and test metrics at those fixed operating points: Test error (EER) with FAR@EER/FRR@EER, and Acc/F1/FRR @ 1% FAR.

1) CNN-ONLY (IMAGE)

Deep architectures demonstrate superior feature representation for writer verification. VGG-16-BN and ResNet-18 achieve low validation EERs (≈ 0.41% and 0.65%) and very strong Acc @1% FAR (≥ 99.24%) with FRR @1% FAR ≤ 0.27%. By contrast, DeepWriterID shows a higher error (Val EER ≈ 2.73%) and degrades at the 1% - FAR operating point (Acc @1%FAR ≈ 92.75%, FRR ≈ 16.59%).

2) RNN-ONLY (SEQUENCE)

Sequence-only encoders are not competitive at low FAR: AUC ≈ 89 − 93%, Val EER 14.7-19.9%, Acc@ 1%

FAR $\approx$ 63 $-$ 73%, and FRR@1% FAR $\approx$ 67 $-$ 83%, indicating that temporal cues alone are insufficient.

3) HYBRID CNN+RNN

Fusing spatial and temporal cues significantly improves performance. KhmerWriterID attains AUC $\approx$ 99.92%, Val EER $\approx$ 0.32%, Acc @1%FAR $\approx$ 99.27%, and FRR@1% FAR $\approx$ 0.07%, outperforming other hybrids (e.g., BiLSTM + SimpleCNN, LSTM + AlexNet).

VI. DISCUSSION AND CONCLUSION

A. DISCUSSION

This work addresses writer verification (1:1) for online Khmer handwriting using Siamese neural networks under writer-disjoint train/validation/test splits. Operating thresholds are selected once on the validation set the equal-error threshold τ_{EER} and the strict operating point $\tau_{\text{FAR}} = 1\%$ and then frozen for test. We report ROC-AUC (threshold-free), EER, FAR/FRR_EER, and Acc/F1/FRR_FAR=1%.

1) RNN-ONLY (STROKE-SEQUENCE) ENCODERS

Sequence-only models are insufficient under stringent low-FAR operation: AUC $\approx$ 89 $-$ 93%, validation EER $=$ 14.7 $-$ 19.9%, and Acc@FAR $=$ 1% $\approx$ 63 $-$ 73% with FRR @ FAR $=$ 1% $\approx$ 67 $-$ 83%. Temporal dynamics alone (pen-state transitions, stroke order, and rhythm) are sensitive to trajectory noise and writing-speed variability, making it difficult to satisfy a 1% FAR constraint.

2) CNN-ONLY (IMAGE) ENCODERS

Deep architectures demonstrate strong feature representation for writer verification. VGG-16-BN and ResNet-18 achieve low validation EERs of 0.41% and 0.65%, respectively, with Acc @ FAR $=$ 1% $\geq$ 99.24% and FRR @ FAR $=$ 1% $\leq$ 0.27%. In contrast, DeepWriterID shows higher error (validation EER $=$ 2.73%) and degrades at the strict operating point (Acc@FAR $=$ 1% $=$ 92.75%, FRR@FAR $=$ 1% $=$ 16.59%), indicating weaker performance in the low-FAR regime.

3) HYBRID CNN+RNN ENCODERS

Fusing spatial and temporal cues markedly improves robustness. The proposed KhmerWriterID attains AUC $=$ 99.92%, validation EER $\approx$ 0.32%, Acc@FAR $=$ 1% $=$ 99.27%, and FRR@FAR $=$ 1% $\approx$ 0.07%. It also outperforms other hybrid variants (e.g., CNN + BiLSTM and CNN+LSTM baselines), which remain competitive but exhibit higher FRR at FAR $=$ 1%. The improvement arises from complementarity: the CNN branch captures glyph shape and consonant-vowel-diacritic stacking patterns, while the sequence branch captures stroke order and pen-lifts. This dual-stream design distinguishes writers who produce similar visual forms via different stroke dynamics and remains resilient when either spatial or temporal evidence is noisy.

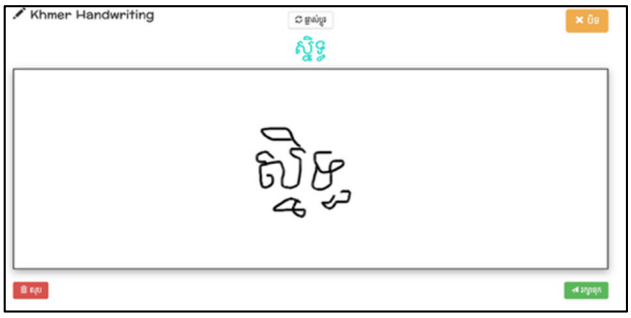

FIGURE 9. User interface for drawing Khmer scripts using OTKH. The black word (ស្ដី) represents the user's handwritten input, while the blue word above represents the word generated from the database (photograph by author). The word drawn by the user is represented as a series of points (x,y).

B. LIMITATION

This study evaluates verification (1:1) only; we do not evaluate closed-set or open-set identification (1:N) and therefore do not report CMC curves. Domain shift (device, stylus, acquisition platform) and cross-dataset generalization remain to be quantified. While results at FAR $=$ 1% are strong, post-hoc calibration and score fusion across multiple words/sessions may further reduce FRR in
operational settings. Finally, on-device efficiency (compression/ pruning/ quantization) and fairness across writer demographics remain important directions for future work.

C. CONCLUSION

We reframed online Khmer handwriting as a writer-verification (1:1) problem and evaluated three Siamese model families RNN-only (stroke sequence), CNN-only (image), and hybrid CNN+RNN under strict writer-disjoint splits using two validation-selected thresholds that are frozen for test evaluation: τ_{EER} and $\tau_{\text{FAR}=1\%}$. Across all settings, the proposed KhmerWriterID (dual-stream CNN + BiGRU) consistently performs best, achieving AUC $=$ 99.92%, EER $=$ 0.25 $-$ 0.32%(val/test), Acc @FAR $=$ 1% $=$ 99.27%, and FRR @FAR $=$ 1% $\approx$ 0.07%. The results indicate a consistent performance hierarchy (RNN-only $<$ CNN-only $<$ Hybrid) and confirm that fusing spatial glyph cues (shape and stacking) with temporal dynamics (stroke order and pen-lifts) is critical for robust, low-FAR verification in Khmer script. Practically, these findings indicate that KhmerWriterID supports verification scenarios requiring strict false-accept constraints while keeping false rejections low, which is desirable for biometric access control and forensic screening.

VII. FUTURE WORK

Future work will extend KhmerWriterID beyond pairwise verification to deployment-oriented settings. First, although this paper evaluates verification only, we will add a few-shot (K-shot) enrollment protocol e.g., k $\in \{1, 3, 5\}$ that

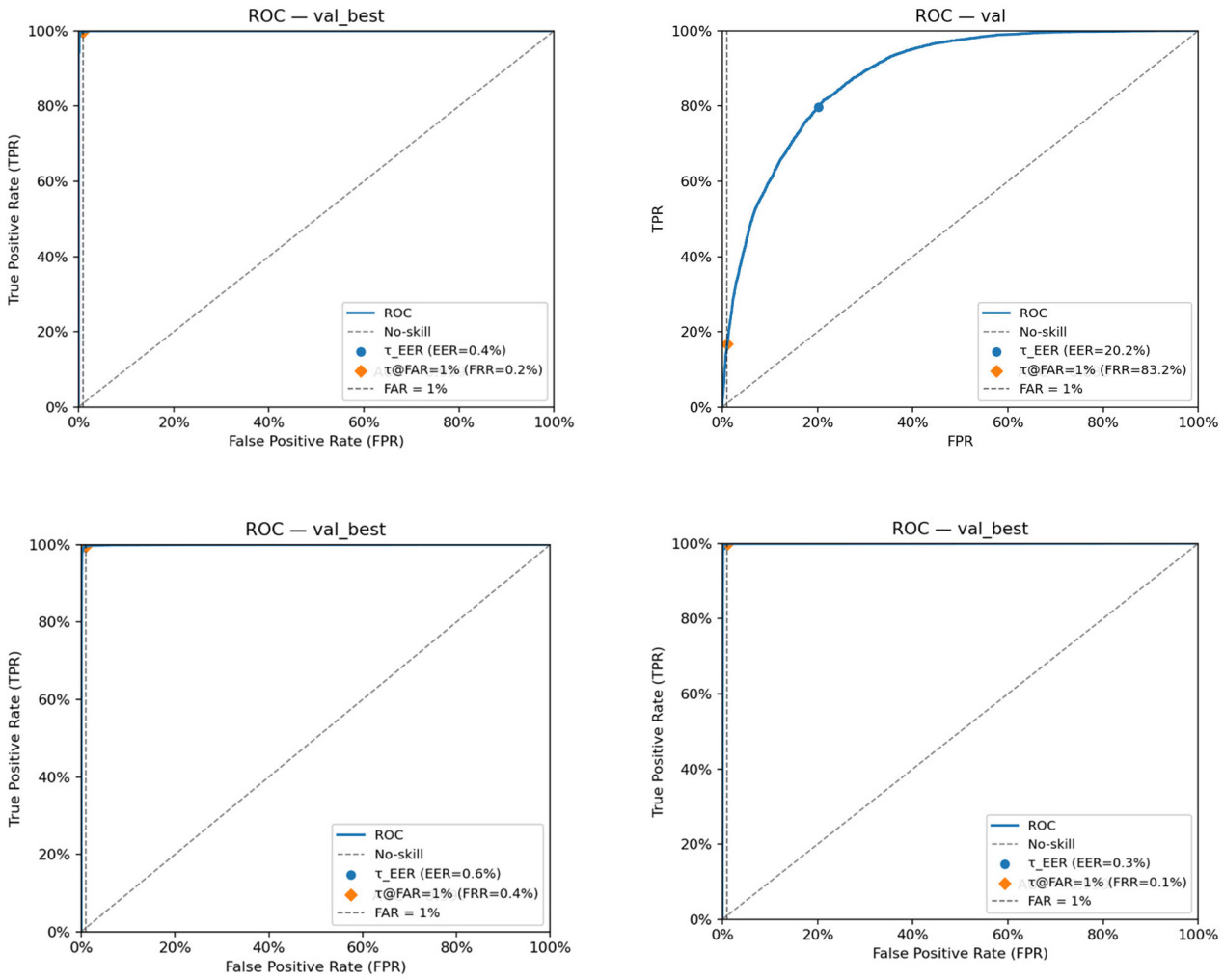

FIGURE 10. ROC curves for four representative models, VGG-16-BN (CNN), BiLSTM (RNN), LSTM+ SimpleCNN (hybrid), and KhmerWriterID (proposed) at the best validation epoch (val_best). Filled circle (●) marks the validation-selected. τE E R; filled diamond (♦) marks the validation-selected $\tau FAR = 1\%$. The vertical dashed line indicates FAR = 1%; the diagonal shows the no-skill classifier.

averages L 2 -normalized embeddings into a re-normalized template and scores probes by cosine similarity; the operating threshold will be fixed on the validation split at FAR = 1% from the impostor probe→ template distribution and applied unchanged to test, reporting FRR@FAR = 1% versus K. We will also study cross-device and cross-session generalization with heterogeneous hardware and sessions; expand the benchmark with more writers, words, and conditions; and complement verification with closed-set identification (CMC/Top-k) and open-set evaluation. On the modeling side, we plan to explore stronger objectives (ArcFace/CosFace/supervised contrastive), self-supervised pretraining, Transformer-style temporal encoders, and alternative fusion schemes, alongside efficiency work (pruning, quantization, distillation) with reports of parameters, FLOPs, latency, and energy. Robustness and calibration under distribution shift (z-/t-norm, temperature scaling, uncertainty) will be investigated, as will integration with character/diacritic segmentation to disentangle content from writer traits. Finally, we will formalize privacy, security, and fairness checks, and adopt human-in-the-loop data growth (active learning, label-quality control) to continuously improve the dataset and model.

APPENDIX A
TOOL DEVELOPMENT AND DATA COLLECTION

To support the data collection process, an online platform called Online Tools for Khmer Handwriting (OTKH) was developed.

This web-based tool enables users to register, create accounts, and log in to write Khmer words directly via a digital interface. The system records each user's handwriting behavior by capturing stroke trajectories in the form of (x, y) coordinate sequences, along with associated user metadata. While the visual appearance of handwriting produced on the platform may differ slightly from that written on paper

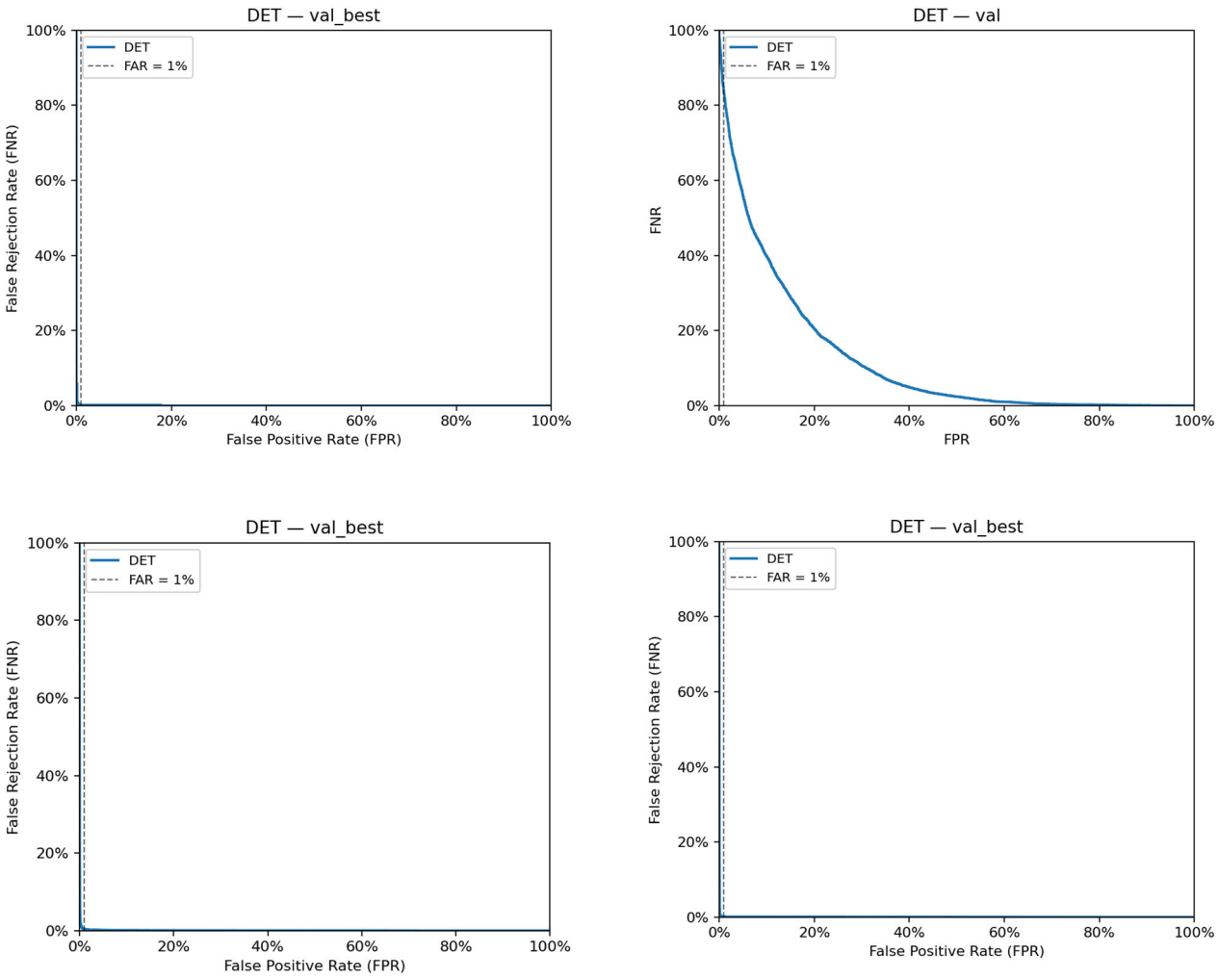

FIGURE 11. DET curves (FNR vs. FPR, log-scaled axes style) for VGG-16, BiLSTM, LSTM+SimpleCNN, and KhmerWriterID at val_best. The vertical dashed line marks FAR = 1%. These plots complement the ROC view by expanding the low-error region around the 1% FAR operating point used in the paper.

due to the lack of tactile feedback, pressure sensitivity, and screen surface differences, the core aspects of the writing style are typically preserved. These patterns are shaped by the user's intrinsic motor habits, which remain consistent across different writing environments.

However, the validity and reproducibility of handwriting data collected through the OTKH platform may be affected by several variables, including the type of input device, screen size, and the user's level of familiarity with the interface. Initially, users may encounter a learning curve when transitioning to a digital handwriting environment, potentially resulting in the increased variability in early samples. As users become more comfortable with the platform over time, their writing behavior tends to stabilize, thereby improving data consistency. Moreover, the frequency of data entry can influence the uniformity of handwriting; while repeated interactions with the tool may lead to greater consistency due to increased familiarity, external factors such as fatigue may still introduce subtle variations in stroke patterns.

To develop the OTKH platform, Python a widely adopted high-level, general-purpose programming language introduced by programming language (van Rossum, 1991) in an opensource a Python-based [26] was used in conjunction with Django [27], an open-source Python-based web framework designed to streamline web application development and minimize redundant coding tasks. Django provides a modular structure that supports rapid development and deployment by leveraging pre-built components. As noted by Adrian Holovaty et al. [28], Django follows the Model-View-Template (MVT) architectural pattern, which is conceptually aligned with the Model-View-Controller (MVC) design framework. The MVT architecture separates application logic into three main components:

- **Model:** Responsible for managing the data layer, the model defines the structure of the data and facilitates

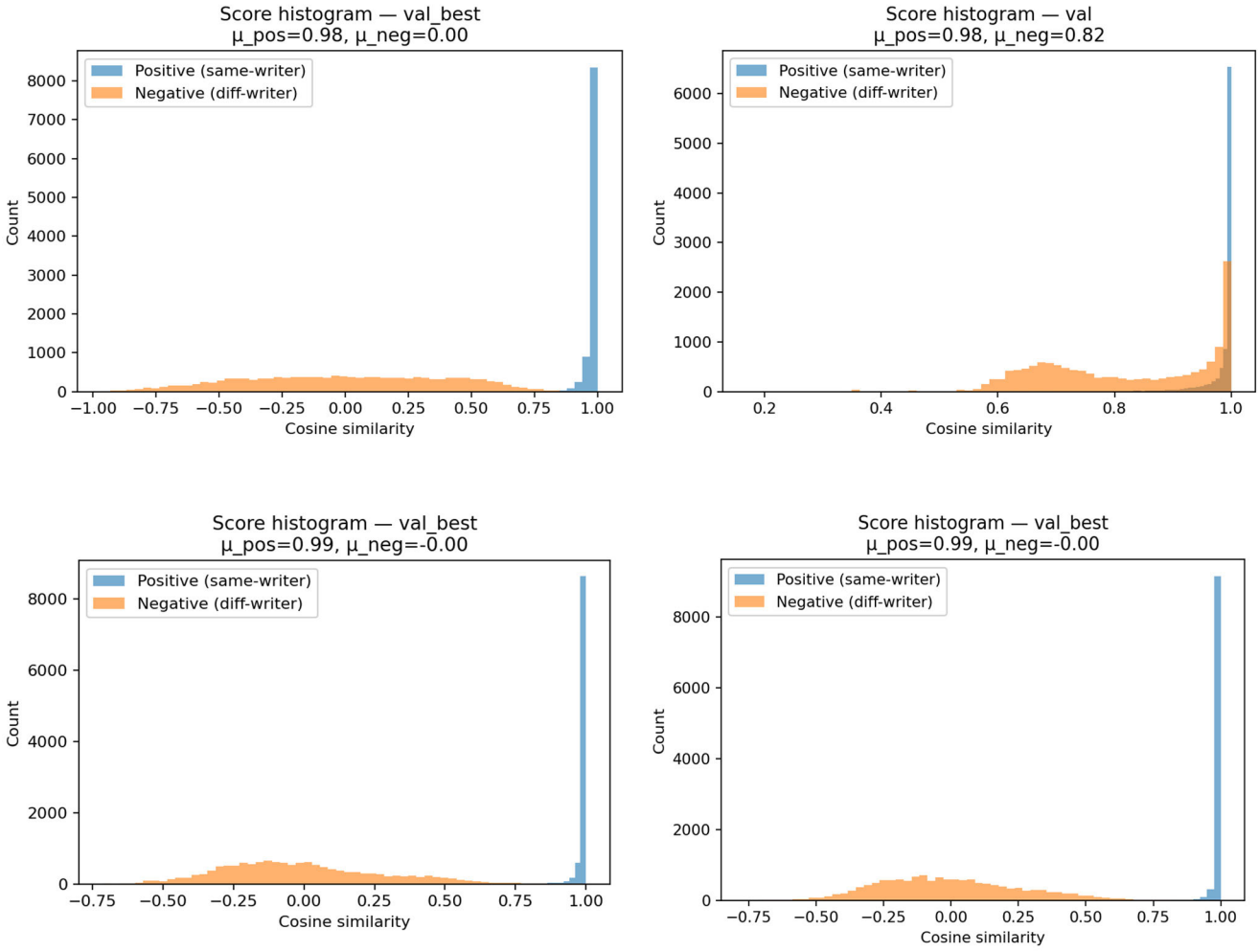

FIGURE 12. Cosine-similarity score histograms for genuine (same-writer) and impostor (different-writer) pairs at val_best for VGG-16, BiLSTM, LSTM+SimpleCNN, and KhmerWriterID. Mean scores. μ_{pos} and μ_{neg} shown above each panel. Larger separation and a tighter positive peak indicate stronger verification margins at the chosen operating thresholds.

saving user input to the database. It handles the logical organization of the application and determines how information is stored and retrieved.

- **View:** This component acts as an intermediary between the model and the template, processing data, and user input. In Django, views handle request processing and control the application's logic, though their role differs slightly from views in traditional MVC patterns.
- **Template:** Serving as a presentation layer, the template renders dynamic content for display to users. It handles how information is presented in the user interface. When a request is made, the controller (implicitly handled by Django) determines the appropriate view and returns a rendered template as a response.

The user interface for drawing Khmer scripts using the OTKH platform (Fig. 9) allows users to draw Khmer words while logged into their accounts. In total, we collected 4,878 instances of handwritten words from 298 participants, with each participant contributing an average of about 16 word samples. The resulting database includes stroke coordinate sequences, word labels, and user demographic attributes, providing a robust foundation for writer verification research.

APPENDIX B
ROC GRIDS FOR ALL MODELS. VALIDATION ROC FOR THE FULL SET OF CNN-ONLY, RNN-ONLY, AND HYBRID SIAMESE MODELS; MARKERS DENOTE τ_{EER} AND $\tau_{FAR=1\%}$

See Figure 10.

APPENDIX C
DET CURVES (SAME MODEL SETS). VALIDATION DET FOR THE SAME CONFIGURATIONS; DASHED VERTICAL LINE INDICATES FAR = 1%

See Figure 11.

APPENDIX D
SCORE HISTOGRAMS. POSITIVE (SAME-WRITER) VS. NEGATIVE (DIFFERENT-WRITER) SIMILARITY DISTRIBUTIONS AT THE VALIDATION-BEST CHECKPOINT FOR REPRESENTATIVE MODELS; TITLES ANNOTATE μ_{POS} AND μ_{NEG}

See Figure 12.

DATA AVAILABILITY

The KhmerWriterID dataset is not publicly released due to privacy and consent restrictions. It can be made available from the corresponding author upon reasonable request for research purposes. The source code and trained model checkpoints can also be provided by the corresponding author upon reasonable request.

REFERENCES

[1] M. Javidi and M. Jampour, "A deep learning framework for text-independent writer identification," *Eng. Appl. Artif. Intell.*, vol. 95, Oct. 2020, Art. no. 103912.

[2] L. Zhiyuan, J. Min, W. Qi, and L. Huaxiang, "Deep template matching for offline handwritten Chinese character recognition," *J. Eng.*, vol. 2020, no. 4, pp. 120–124, Apr. 2020.

[3] N. Kimlong and V. Dona, "Online handwriting identification from Khmer script using a deep learning approach with KhNet," in *Proc. CITA*, 2025, pp. 751–763.

[4] M. Javidi and M. Jampour, "Handwriting analysis: Psychological aspects and pattern recognition," *Pattern Anal. Appl.*, vol. 23, no. 4, pp. 899–912, Nov. 2020.

[5] C. C. Tappert, C. Y. Suen, and T. Wakahara, "The state of the art in online handwriting recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 12, no. 8, pp. 787–808, Aug. 1990.

[6] L. Likforman-Sulem, A. Esposito, M. Faundez-Zanuy, S. Clemençon, and G. Cordasco, "EMOTHAW: A novel database for emotional state recognition from handwriting," 2022, *arXiv:2202.12245*.

[7] V. K. Seema and R. Shilpa, "Recognition of emotional state based on handwriting analysis and psychological assessment," *Int. J. Eng. Adv. Technol.*, vol. 8, no. 6, pp. 4395–4402, Aug. 2019.

[8] C. E. Chaski, "Who's at the keyboard? Authorship attribution in digital evidence investigations," *Int. J. Digit. Evid.*, vol. 4, no. 1, pp. 1–13, 2005.

[9] M. A. M. Hasan, J. Shin, and M. Maniruzzaman, "Online kanji characters based writer identification using support vector machine," *Appl. Sci.*, vol. 12, no. 20, p. 10249, Oct. 2022.

[10] C. Adak, B. B. Chaudhuri, and M. Blumenstein, "An empirical study on writer identification and verification from intra-variable individual handwriting," *IEEE Access*, vol. 7, pp. 24738–24758, 2019.

[11] Z. Y. He and Y. Y. Tang, "Chinese handwriting identification using convolutional neural networks," *IEEE Trans. Cybern.*, vol. 54, no. 2, pp. 678–690, Feb. 2024.

[12] B. Purkaystha, T. Datta, and M. S. Islam, "Bengali handwritten character recognition using deep convolutional neural network," in *Proc. 20th Int. Conf. Comput. Inf. Technol. (ICCIT)*, Dec. 2017, pp. 1–5.

[13] B. Annanurov and N. M. Noor, "Handwritten Khmer text recognition," in *Proc. IEEE Int. WIE Conf. Electr. Comput. Eng. (WIECON-ECE)*, Dec. 2016, pp. 176–179.

[14] R. Plamondon and S. N. Srihari, "Online and off-line handwriting recognition: A comprehensive survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 1, pp. 63–84, Jan. 2000, doi: 10.1109/34.824821.

[15] M. Bulacu and L. Schomaker, "Text-independent writer identification and verification using textural and allographic features," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 4, pp. 701–717, Apr. 2007.

[16] S. Dey, A. Dutta, J. Ignacio Toledo, S. K. Ghosh, J. Llados, and U. Pal, "SigNet: Convolutional Siamese network for writer independent offline signature verification," 2017, *arXiv:1707.02131*.

[17] K. Ngin, "A deep learning approach for identifying individuals based on their handwriting," *Techno-Sci. Res. J.*, vol. 13, no. 1, pp. 52–58, Apr. 2025.

[18] L. Mengsay. (2019). *Handwritten Khmer Digit Recognition*. Accessed: Jun. 15, 2020. [Online]. Available: https://medium.com/towards-data-science/handwritten-khmer-digit-recognition-860edf06cd57

[19] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah, "Signature verification using a Siamese time delay neural network," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 1994, pp. 737–744.

[20] W. Yang, L. Jin, and M. Liu, "DeepWriterID: An end-to-end online text-independent writer identification system," *IEEE Intell. Syst.*, vol. 31, no. 2, pp. 45–53, Mar. 2016.

[21] A. Graves, *Supervised Sequence Labelling With Recurrent Neural Networks*, vol. 385. Berlin, Germany: Springer, 2012.

[22] D. Sounak, K. B. Ayan, S. Palaiahnakote, P. Umapada, and P. R. Partha, "Signet-bengali and signet-indic: Two benchmark datasets for offline signature verification in indic scripts," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 9, pp. 4717–4731, Sep. 2022.

[23] H. Sasagawa, "The establishment of the national language in twentieth-century Cambodia: Debates on orthography and coinage," *Southeast Asian Stud.*, vol. 4, no. 1, pp. 43–72, Apr. 2015.

[24] J.-M. Filippi and C. Vatho. (2016). *The Linguistic Atlas of Cambodia and Standard Khmer Language*. [Online]. Available: https://www.researchgate.net/publication/359083929_The_Linguistic_Atlas_of_Cambodia_and_Standard_Khmer_Language

[25] M. Kesiman, D. Valy, J.-C. Burie, E. Paulus, M. Suryani, S. Hadi, M. Verleysen, S. Chhun, and J.-M. Ogier, "Benchmarking of document image analysis tasks for palm leaf manuscripts from Southeast Asia," *J. Imag.*, vol. 4, no. 2, p. 43, Feb. 2018.

[26] Python Software Foundation. *Python Language Reference*. Accessed: Aug. 3, 2024. [Online]. Available: https://docs.python.org/3/reference/index.html

[27] P. Thakur and P. Jadon, "Django: Developing web using Python," in *Proc. 3rd Int. Conf. Advance Comput. Innov. Technol. Eng. (ICACITE)*, May 2023, pp. 303–306.

[28] Django Software Foundation. *Django Web Framework Documentation*. Accessed: Aug. 3, 2024. [Online]. Available: https://docs.djangoproject.com/en/5.0/faq/general/

KIMLONG NGIN received the B.S. degree in computer science from the Royal University of Phnom Penh, Cambodia, in 2013, and the M.S. degree in computer science (applied computer applications) from the University of Science and Technology of China (USTC), Anhui, China, in 2019. He is currently pursuing the Ph.D. degree with the Institute of Technology of Cambodia (ITC), Phnom Penh. His research interests include computer vision, natural language processing, and machine learning, with a focus on handwriting recognition, writer verification, and Khmer language processing.

DONA VALY received the Ph.D. degree in engineering science and technology from the Université Catholique de Louvain (UCLouvain), Belgium. He is currently the Head of the Mechatronics and Information Technology Research Unit, Institute of Technology of Cambodia (ITC), Phnom Penh, Cambodia. His research interests include computer vision and natural language processing. He has been active in academia, since 2011, with experience in teaching and conducting research in these areas.

SOKKHEY PHAUK is currently the Head of the Department of Applied Mathematics and Statistics and a Program Coordinator of the master of data science with the Institute of Technology of Cambodia (ITC), Phnom Penh, Cambodia. He leads the Research and Data Analytics (ReDA) Laboratory, where he supervises multidisciplinary work on educational data mining, predictive learning analytics, and AI-based monitoring frameworks for inclusive education. His research interests include machine learning, natural language processing, and large language models, with applications aligned with the sustainable development goals (SDGs), particularly SDG 4 (quality education), SDG 3 (good health and well-being), and SDG 9 (industry, innovation, and infrastructure).

VANNARO PIN received the B.S. degree in agricultural science from Veat Moharusey University, the M.S. degree in agricultural science from the Chamroeun University of Poly-Technology, and the Ph.D. degree in education in Cambodia. He is currently a Rector with the University of Heng Samrin Tboung Khmum (UHST), Cambodia. Previously, he was a Vice Rector with the Chea Sim University of Kamchaymear. He has more than two decades of experience in higher education leadership and applied research in rural and agricultural development. He has participated in collaborative projects supported by international organizations. His interests include academic leadership, curriculum development, and sustainable farming systems.

BAMNANG YIN was born in Phnom Penh, Cambodia, in 1980. He received the B.S. degree in computer science from Norton University, Phnom Penh, in 2003, and the M.S. degree in business information technology from Northumbria University at Newcastle, Malaysia, in 2008. From 2003 to 2006, he was an IT Executive at P&S Construction. From 2008 to 2009, he was an IT Technical Support Engineer at G4S Security Services, Cambodia. Then, he spent seven years as a Lecturer in information technology at several universities in Cambodia. In 2016, he joined the University of Heng Samrin Tboung Khmum (UHST), where he was a Lecturer and the Deputy Director. Since 2022, he has been the Director of the Institute of Information Technology, UHST.

SOKROEURN ANG (Member, IEEE) received the M.S. degree in cybersecurity from the Royal Holloway, University of London, U.K. He is currently pursuing the Ph.D. degree in cybersecurity with Lincoln University College, Malaysia. He completed a MicroMasters Program in Cybersecurity with Rochester Institute of Technology (RIT), Rochester, NY, USA. His certifications include CISSP, CISA, CISM, CEH, and AWS Certified Cloud Practitioner. He is currently a Senior Lecturer in cybersecurity. Since 2015, he has been teaching ICT and cybersecurity and has more than ten years of professional experience across central banking, private banking, and internet service providers. His expertise includes cybersecurity risk assessment, IT governance, network and web application security, incident response, business continuity and disaster recovery planning (BCP/DRP), cloud security, vulnerability assessment and penetration testing (VAPT), and IT auditing.