

Beyond the Image: Inferring Latent Signature Dynamics with DynaSigNet for Robust Writer Identification

SK GOLAM SAROWAR HOSSAIN¹, MRIDUL GHOSH^{2,*}, ASIFUZZAMAN LASKAR¹
TONMOY METE⁴, MARIA ZDIMALOVA⁵, KAUSHIK ROY³
SK MD OBAIDULLAH¹

¹Department of Computer Science and Engineering
Aliah University
Kolkata 700160, West Bengal
INDIA

²Department of Computer Science
Shyampur Siddheswari Mahavidyalaya
Howrah 700312, West Bengal
INDIA

³Department of Computer Science
West Bengal State University
Barasat 700126, West Bengal
INDIA

⁴Department of Computer Science
Asutosh College
Kolkata 700026, West Bengal
INDIA

⁵Department of Mathematics and Descriptive Geometry
Slovak University of Technology in Bratislava
SLOVAKIA

Abstract: - Signature-based writer identification from static images is a critical biometric task, yet it traditionally ignores the rich, dynamic information of the signing process. This paper introduces DynaSigNet (Dynamic Signature Network), a novel deep learning framework that overcomes this limitation by learning to infer the latent generative dynamics—trajectory, velocity, and pressure—from a single offline signature. Our architecture employs a synergistic CNN-Transformer backbone to produce a holistic feature representation that captures both local textures and global structural dependencies. The core of our method is a multi-task learning paradigm where this shared representation is decoded in parallel: a primary head for writer classification, and auxiliary motor-imagination heads for predicting the dynamic properties. By training the model to understand how the signature was formed, we achieve a more discriminative feature space. Validated on a challenging bilingual dataset of Roman and Bengali scripts, DynaSigNet achieves state-of-the-art performance, demonstrating that inferring latent motor processes is a powerful new direction for static biometric analysis. A system graphical user interface (GUI) is also implemented.

Key-Words: Signature identification, Deep learning, Visual transformer, Latent Dynamics, Biometrics, Dynamic Signature Network, Multi-script recognition

Received: April 7, 2026. Revised: June 11, 2026. Accepted: July 13, 2026. Published: October 12, 2026.

1 Introduction

Among the numerous biometric methods employed for personal identification, signatures stand out as one of the most widely recognized and utilized in everyday life. Whether in banking, corporate offices, or formal institutions, signature-based authentication re-

mains a trusted practice. Despite the advancement of other biometric systems [1] such as facial recognition, fingerprint scanning, iris identification, vein pattern analysis, and DNA testing, signatures continue to be favored due to their simplicity and traditional usage. Consequently, researchers have dedicated decades to

developing automated systems for writer identification using signature analysis [2], [3], [4]. Each individual's signature is distinct, making it an effective marker for identification. Historically, human experts analyzed handwriting manually, but with the evolution of machine learning and deep learning techniques [5], [6], [7], automated systems have emerged, aiming to enhance precision, minimize human error, and lower false-positive rates by combining computational analysis with expert validation.

This paper presents a method for analyzing handwritten documents by retrieving digitized images based on their graphical content, specifically handwriting styles and authorship. It outlines two main applications: analyzing the textual content, which requires transcription before applying information retrieval techniques, though this is currently limited by the inefficiencies of handwriting recognition technology; and analyzing the graphical content, where documents are retrieved based on handwriting styles to identify specific calligraphy, detect multiple writing styles, or date documents in relation to an author's chronology. The proposed approach leverages recent advancements in automatic handwriting analysis for these tasks. Automated writer identification systems offer multiple advantages. They provide easy access to extensive datasets, enabling detailed evaluation of both text-based and graphical features in handwritten documents. These systems can identify individuals by analyzing the consistency and unique characteristics of their handwriting, often surpassing traditional methods like facial or fingerprint recognition in speed. Additionally, they boost security, reduce operational expenses, and offer a secure, permanent method for database storage.

The primary objectives of a writer identification system are to identify the most efficient techniques for accurate identification, improve the system's accuracy, and extend its use to forensic and historical document analysis. Another key goal is the conversion of handwritten documents into digital formats, ensuring systematic processes for multi-writer applications, and recognizing individuals through the unique traits of their handwriting. These aims must be achieved while maintaining both computational efficiency and cost-effectiveness, which encourages continued research and development in this field.

The innovative application of pattern recognition in writer identification has significant potential and presents numerous challenges, especially when compared to other biometric systems. Advances in this area could place writer identification at the forefront of pattern recognition technology. Key areas of application include forensic document examination, criminal justice, biometric security systems, personal identification, secure mobile banking, network access

control, property protection, legal document verification, signature authentication, and financial transaction security. Additionally, the system can play a crucial role in archiving historical documents, allowing for effective indexing and retrieval. The broad applicability of writer identification systems highlights their versatility and underscores their potential to improve security and operational efficiency in multiple industries.

Signature identification [8], [9], [10] can be divided into offline (static) and online (dynamic) categories. In offline signature identification, writing is done on solid paper, and the signature is captured using a scanner or camera. The system then examines the topology and contour of the digitized signature. It is static, as it does not require stream processing. On the other hand, online signature identification captures live input by means of devices like tablets or smartphones. This can include a stylus or pen, a touch-sensitive surface, and software that tracks the movement and pressure of the stylus. Offline identification is more challenging to analyze because it is limited to a static two-dimensional image, while online identification is easier because it captures dynamic features of the writing process.

In the present study, we focus on Bangla and English languages that correspond to Bangla and Roman scripts. These are not two different languages; they are two entirely different script systems two systems that are most commonly used in India and further. Bangla (or Bengali) is the official language of Bangladesh and is also widely spoken in the Indian state of West Bengal. Compared to English which serves as an umbrella language across the globe and across sectors like that of startups, education, and governance. These two languages, as examples, exhibit a range of scripts that an effective writer identification system should incorporate, as they belong to different families of writing systems. For instance, Bangla is an Abugida writing system with a connecting "matra" while English is a true alphabetic writing system with a much simpler and more linear structure. This study seeks to develop an advanced system for identifying writers through signatures that are written in different languages. The diagram in Figure 1 illustrates the process of training a system and identifying writers based on their Bangla and Roman signatures. The contribution of this work can be summarized as:

- We introduce a new paradigm for offline writer identification by framing it as a problem of inferring the latent generative motor process (trajectory, velocity, pressure) from a static 2D image, shifting the analysis from static pattern recognition to dynamic process modeling.
- We propose DynaSigNet, a novel multi-task ar-

chitecture featuring a shared perception backbone and parallel decoders for simultaneous writer identification and dynamic process reconstruction, which forces the learning of a more discriminative feature space.

- We develop a novel heuristic framework to generate pseudo-ground truth for dynamic properties (trajectory, velocity, pressure) directly from static images, enabling the training of dynamic inference models on standard offline signature datasets.
- We provide empirical validation of the complete framework on a challenging bilingual (Roman and Bengali) dataset, establishing a new state-of-the-art in script-invariant writer identification and proving the efficacy of our proposed dynamic inference approach.

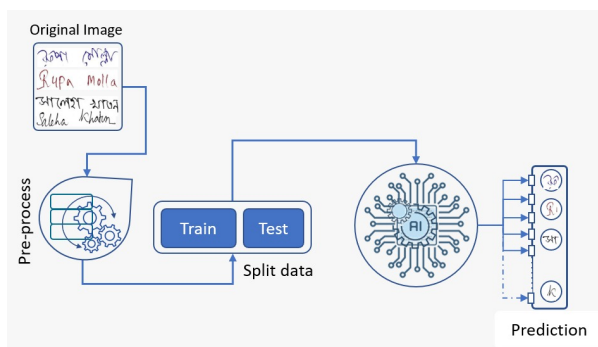


Fig. 1: Overview of the system for writer identification using handwritten signatures as input.

Source: created by the authors.

The remainder of this paper is organized as follows: Section 2 discusses the literature review. Section 3 details the methods, including data preprocessing and the proposed architecture. Experimental details, including the dataset, are discussed in Section 4, followed by the results in Section 5 and the conclusion in Section 6.

2 Literature study

Signature-based writer identification plays a vital role in forensic investigations and document verification. By analyzing distinctive handwriting characteristics, this method ensures accurate authorship attribution, thus strengthening the credibility of evidence in legal proceedings and improving security measures.

Document image processing, particularly in a multi-script or multilingual environment, requires a stage of writer identification and verification using handwritten signature pictures. So far, several techniques have been devised to address this problem.

Support vector machine classifiers were exploited for offline signature identification and verification on an English dataset using global, directional, and grid features, [11]. The Persian and English signature database using Contourlet Transform was discussed, [12], whereas in [13], both the contourlet transform and support vector machine were used to identify and verify the Persian Database using global geometric and moment-based features. The study on Arabic handwriting recognition using the optimization of MCS based on diversity measures was conducted, [14]. In [15], foreground and background features for the same were discussed. In [16], [17], work was conducted on English and Hindi signature dataset GPDS using modified gradient features, Gabor filters, SURF features, water reservoir features, loop features, aspect ratios, etc. A two-stage approach was used, [18], to work on a multi-script signature image dataset (Bangla, Hindi, and English) GPDS160 using global, grid, and texture features.

A script level recognition system was suggested for six handwriting Indian scripts: Bangla, Devanagari, Gurmukhi, Malayalam, Oriya, Telugu and the Roman script, [19]. In a character recognition problem of moderately large classes, [20], a fair degree of training was accomplished on a 5-layer CNN, among other things, in addition to other training parameters. The methodology employed, [21], on efficient signature verification works with an interval symbolic representation and a fuzzy similarity measure applied to both English signature images (GPDS-300) and the images of the Bengali and Hindi signatures (BHSig260). In [22], learning features for offline handwritten signature verification using deep convolutional neural networks were proposed to address the difficulty of obtaining good features and improve system performance.

A DAG-CNN for offline signature verification, focusing on classifying and verifying the authenticity of offline signatures using a writer-independent method, was discussed, [22]. In [10], a two-stage convolutional neural network (CNN) architecture for signature-based writer identification was proposed, utilizing a stack ensemble network (SEN) that combines truncated versions of VGG-19 and EfficientNet-B0 with a shallow base CNN model. Previous research in this area has primarily utilized learning-based approaches to detect writing patterns and feed them into machine learning models for writer determination. In a different context, [23], sensor fusion and multimodal learning for robotic grasp verification were explored using neural networks. This involved multimodal data fusion techniques such as early, late, and intermediate fusion with 3D convolution-based multimodal neural networks.

In [24], the concept of conditional multimodal bio-

metrics (CMB) and a corresponding learning model trained with N multimodal biometrics, showcasing a generalized approach to multimodal biometrics, were introduced. In [25], a novel data augmentation method called Inverted Cutout was proposed to enhance the occlusion robustness of CNNs during training. In [26], a method for writer retrieval using compact convolutional transformers (CCTs) and Net-MVLAD was presented, demonstrating superior performance compared to traditional CNNs for writer retrieval tasks. Offline handwriting image analysis for predicting Alzheimer's disease through deep learning was explored, [27], emphasizing the importance of extracting features directly from handwritten images. In a different domain, [28], PRNU-Net was developed as a deep learning approach for source camera model identification based on videos captured with smartphones. In [29], BAFL (Federated Learning for Base Ablation) was proposed as a cost-effective communication strategy in federated learning, aiming to reduce communication costs while improving weight initialization. A unified statistical model was explored, [30], for multi-band texture representation using finite mixtures of multivariate generalized Gaussian distributions. In general, the review of the literature highlights the diverse applications of neural networks, multimodal learning, data augmentation techniques, and deep learning approaches in various domains such as writer identification, robotic grasp verification, biometrics, occlusion robustness, camera model identification, and texture modeling. These studies collectively contribute to advancing the field of handwritten signature-based writer identification and verification in offline mode.

3 Methodology

The proposed methodology introduces a novel framework, Dynamic Signature Network (DynaSigNet), for robust writer identification. The approach is predicated on the hypothesis that a model trained to infer the latent dynamic motor processes from a static signature image will develop a more discriminative feature representation. This section details the complete pipeline, beginning with the crucial data pre-processing and augmentation steps, followed by the architecture of the multi-task model and its training objective.

3.1 Data Pre-processing and Augmentation

To ensure model stability, robustness, and generalization, a comprehensive pre-processing and augmentation pipeline is applied to each input signature image.

3.1.1 Rescaling Normalization

Initially, each image, with pixel intensity values in the 8-bit integer range $[0, 255]$, is normalized to a floating-point range of $[0, 1]$. For an original pixel value x , the normalized value x' is calculated as:

$$x' = \frac{x}{255} \quad (1)$$

This normalization is fundamental for stable gradient-based optimization during the training of the deep neural network.

3.1.2 Geometric Data Augmentation

During the training phase, we apply a series of stochastic geometric transformations to the normalized images on-the-fly. This process artificially expands the training dataset and introduces critical invariances, thereby preventing overfitting and enhancing the model's ability to generalize to unseen data. The applied augmentations are defined as follows:

- **Shear Transformation:** Skews the image along the horizontal axis to simulate variations in writing slant. For a pixel at coordinate (x, y) , the new coordinate (x', y') is given by:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & \text{shear_range} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \quad (2)$$

where `shear_range` is a randomly sampled intensity factor.

- **Zoom Augmentation:** Randomly zooms the image by a factor sampled from $[1 - \text{zoom_range}, 1 + \text{zoom_range}]$, where we set `zoom_range` = 0.2. This makes the model robust to variations in signature scale.
- **Horizontal and Vertical Flips:** The image is flipped along its horizontal and vertical axes, each with a 50% probability. For an image of dimensions (width, height), the transformations are:

$$x' = \text{width} - x \quad (\text{Horizontal Flip}) \quad (3)$$

$$y' = \text{height} - y \quad (\text{Vertical Flip}) \quad (4)$$

- **Width and Height Shifts:** The image is translated randomly along its axes. The transformation is defined by:

$$x' = x + \text{width_shift_range} \times \text{width} \quad (5)$$

$$y' = y + \text{height_shift_range} \times \text{height} \quad (6)$$

where the shift ranges are set to $\pm 20\%$ of the image dimension.

The final augmented and normalized data tensor, denoted as $\mathbf{X} \in \mathbb{R}^{H \times W \times C}$, serves as the input to the DynaSigNet architecture.

3.2 DynaSigNet Architecture

The proposed DynaSigNet illustrated in Figure 2 is a multi-task deep learning framework designed to transcend the limitations of static image analysis for writer identification. The core hypothesis is that a model's ability to identify a writer is significantly enhanced if it is compelled to learn the latent neuro-motor dynamics inherent in the signature's creation. To this end, DynaSigNet is architected to perform writer identification simultaneously with three auxiliary motor-imagination tasks: trajectory reconstruction, velocity estimation, and pressure mapping.

The architecture is formally composed of two primary stages: (1) a **Shared Perception Backbone** that generates a rich, contextualized feature representation of the input signature, and (2) a set of parallel **Multi-Task Decoders**, each specialized for a specific task.

3.3 Shared Perception Backbone

Like the nodule feature encoder that is a powerful part of the feature encoder itself, the backbone is a feature encoder that mixes both the local feature extraction networks of CNNs and the global context representation networks of Transformers in a symbiotic relationship. Let an input signature image be denoted by $\mathbf{X} \in \mathbb{R}^{H \times W \times C}$.

3.3.1 CNN Feature Extractor

The initial stage of the backbone is a CNN designed to distill fine-grained spatial features. It consists of an initial convolutional layer followed by a stack of our novel **Combined Blocks (CBs)** which is shown in Figure 3. Each CB is a computationally efficient residual unit that performs a three-stage convolution strategy. For an input feature map $\mathbf{F}_{in}$ to a CB, the output $\mathbf{F}_{out}$ is defined as:

$$\mathbf{F}' = \text{Conv}_{pw}(\sigma(\text{Conv}_{std}(\sigma(\text{Conv}_{dw}(\sigma(\text{BN}(\mathbf{F}_{in}))))))) \quad (7)$$

$$\mathbf{F}_{out} = \mathbf{F}' + \mathbf{F}_{in} \quad (8)$$

where: Conv_{dw} , Conv_{std} , and Conv_{pw} represent depth-wise, standard, and pointwise (1x1) convolutions, respectively; BN denotes Batch Normalization; and σ is the Swish activation function. The residual connection ensures stable gradient flow. The output of this stage is a low-resolution, high-dimensional spatial feature map, $\mathbf{F}_{cnn} \in \mathbb{R}^{H' \times W' \times C'}$.

3.3.2 Transformer Encoder

To model the long-range dependencies between learned features, $\mathbf{F}_{cnn}$ is passed to a Transformer encoder. The 2D feature map is first "patchified" and flattened into a sequence of N tokens, $\mathbf{P} = \{\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_N\}$, where $N = H' \times W'$. Each token $\mathbf{p}_i \in \mathbb{R}^{C'}$. This sequence is then processed by L

layers of Transformer encoder blocks. Each block applies Multi-Head Self-Attention (MHSA) and a Feed-Forward Network (FFN). The self-attention mechanism is defined as:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}\right) \mathbf{V} \quad (9)$$

where $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$ are the query, key, and value matrices derived from the input sequence $\mathbf{P}$, and d_k is the dimension of the key vectors. The output sequence is then reshaped back into a spatial grid, yielding the final **Contextualized Feature Map**, $\mathbf{Z} \in \mathbb{R}^{H' \times W' \times C'}$, which serves as the shared input for all subsequent decoders.

3.4 Multi-Task Decoders

The contextualized feature map $\mathbf{Z}$ is concurrently fed into four specialized decoders.

3.4.1 Probabilistic Framework for Latent Dynamics

To provide a rigorous mathematical foundation for DynaSigNet, we formalize the inference of the signing process as a probabilistic latent generative model. Let $\mathbf{X} \in \mathbb{R}^{H \times W \times C}$ represent the observed static 2D signature image. We postulate that the static image is the result of a latent generative motor process characterized by continuous dynamic variables: trajectory ($\mathbf{Y}_{traj}$), velocity ($\mathbf{Y}_{vel}$), and pressure ($\mathbf{Y}_{press}$).

Our objective is to maximize the joint conditional probability of the discrete writer identity (y_{id}) and these continuous latent dynamics, given the static image observation $\mathbf{X}$. Assuming that the auxiliary dynamic variables are conditionally independent given the deeply contextualized feature map $\mathbf{Z} = f_{\theta_{backbone}}(\mathbf{X})$, the joint conditional distribution is factorized as follows:

$$P(y_{id}, \mathbf{Y}_{traj}, \mathbf{Y}_{vel}, \mathbf{Y}_{press} | \mathbf{X}; \Theta)$$

$$= P(y_{id} | \mathbf{Z}; \theta_{id}) P(\mathbf{Y}_{traj} | \mathbf{Z}; \theta_{traj}) \quad (10)$$

$$\cdot P(\mathbf{Y}_{vel} | \mathbf{Z}; \theta_{vel}) P(\mathbf{Y}_{press} | \mathbf{Z}; \theta_{press})$$

Each decoder in our architecture functions as a parameter estimator for these specific conditional distributions:

- **Identity Distribution:** The writer classification is modeled as a discrete Categorical distribution parameterized by the Identification Head:

$$P(y_{id} | \mathbf{Z}; \theta_{id})$$

$$= \text{Categorical}(\text{softmax}(\mathbf{W}_{id}\mathbf{Z}_{gap} + \mathbf{b}_{id})) \quad (11)$$

- **Trajectory Distribution (Kinetic):** The sequential trajectory points $\mathbf{c}_t = (x_t, y_t)$ are modeled as an autoregressive conditional Gaussian process. At each time step t , the GRU hidden state $\mathbf{h}_t$ estimates the mean of the continuous spatial coordinates:

$$P(\mathbf{c}_t | \mathbf{c}_{<t}, \mathbf{Z}; \theta_{traj}) = \mathcal{N}(\mathbf{c}_t; \mu_\theta(\mathbf{h}_t), \mathbf{I}) \quad (12)$$

- **Velocity and Pressure Distributions (Rhythm & Emphasis):** The dense 2D velocity and pressure heatmaps are formulated as independent pixel-wise continuous Gaussian distributions mapped from the static image space via transposed convolutions. For a given pixel i in the spatial grid:

$$P(\mathbf{Y}_{vel,i} | \mathbf{Z}; \theta_{vel}) = \mathcal{N}(\mathbf{Y}_{vel,i}; \text{Decoder}_{vel}(\mathbf{Z})_i, \sigma^2) \quad (13)$$

$$P(\mathbf{Y}_{press,i} | \mathbf{Z}; \theta_{press}) = \mathcal{N}(\mathbf{Y}_{press,i}; \text{Decoder}_{press}(\mathbf{Z})_i, \sigma^2) \quad (14)$$

By framing the model through this strict probabilistic lens, the network's training objective can be formally defined as Maximum Likelihood Estimation (MLE). Taking the negative log-likelihood (NLL) of the joint distribution transforms the Categorical probability into the Cross-Entropy loss ($\mathcal{L}_{id}$), and the Gaussian probabilities into Mean Squared Error losses ($\mathcal{L}_{traj}$, $\mathcal{L}_{vel}$, $\mathcal{L}_{press}$). Consequently, minimizing this NLL mathematically derives the weighted composite multi-task loss function ($\mathcal{L}_{total}$) presented in Equation (21).

3.4.2 Identification Head

This head performs the primary task of writer classification. It first reduces the spatial dimensions of $\mathbf{Z}$ into a single feature vector $\mathbf{z}_{gap}$ using Global Average Pooling (GAP). This vector is then passed through a fully connected (Dense) layer with a Softmax activation function to produce the final probability distribution over K writer classes.

$$\mathbf{z}_{gap} = \frac{1}{H'W'} \sum_{i=1}^{H'} \sum_{j=1}^{W'} \mathbf{Z}_{ij} \quad (15)$$

$$\mathbf{y}_{id} = \text{softmax}(\mathbf{W}_{id}\mathbf{z}_{gap} + \mathbf{b}_{id}) \quad (16)$$

where $\mathbf{y}_{id} \in \mathbb{R}^K$ is the predicted probability distribution, and $\mathbf{W}_{id}$, $\mathbf{b}_{id}$ are the learnable weights and biases.

3.4.3 Shared Dynamic Convolutional Decoder

As illustrated in our architectural schematic (Figure 2), instead of employing independent decoders or recurrent sequence generators, the three latent dynamic

heads share a common upstream Convolutional Decoder. This design choice optimizes computational efficiency by allowing the network to learn a shared, high-resolution spatial representation of the signature's motor dynamics.

The contextualized feature map $\mathbf{Z}$ from the Transformer encoder is passed into this Convolutional Decoder, which utilizes transposed convolutional layers to progressively upsample the feature space. This results in a shared dynamic feature map, $\mathbf{H}_{dyn} \in \mathbb{R}^{H \times W \times 32}$.

$$\mathbf{H}_{dyn} = \text{Decoder}_{conv}(\mathbf{Z}) \quad (17)$$

3.4.4 Task-Specific Dynamic Heads (Kinetic, Rhythm & Emphasis)

The shared dynamic feature map $\mathbf{H}_{dyn}$ branches into three specialized prediction heads. Rather than autoregressive sequence generation, we formulate the inference of all dynamic properties—trajectory, velocity, and pressure—as dense spatial regression tasks.

Each head consists of a point-wise (1×1) Convolutional layer followed by a Sigmoid activation function to map the 32-channel representation into the desired output dimensionality, normalized between $[0, 1]$:

- **Trajectory Head (Kinetic):** Reconstructs the 2D spatial vector field of the pen's movement. It maps to a 2-channel output representing the spatial x and y trajectory features:

$$\mathbf{y}_{traj} = \sigma(\text{Conv2D}_{1 \times 1, traj}(\mathbf{H}_{dyn})) \in \mathbb{R}^{H \times W \times 2} \quad (18)$$

- **Velocity Head (Rhythm):** Generates a single-channel spatial heatmap representing the inferred speed of the pen at each pixel:

$$\mathbf{y}_{vel} = \sigma(\text{Conv2D}_{1 \times 1, vel}(\mathbf{H}_{dyn})) \in \mathbb{R}^{H \times W \times 1} \quad (19)$$

- **Pressure Head (Emphasis):** Generates a single-channel spatial heatmap representing the inferred pen pressure exerted on the paper:

$$\mathbf{y}_{press} = \sigma(\text{Conv2D}_{1 \times 1, press}(\mathbf{H}_{dyn})) \in \mathbb{R}^{H \times W \times 1} \quad (20)$$

By predicting dense spatial heatmaps directly from a shared convolutional backbone, the model successfully captures the local spatial correlations between trajectory, velocity, and pressure simultaneously.

3.5 Training Objective and Algorithm

The entire DynaSigNet model, parameterized by θ , is trained end-to-end by minimizing a composite multi-task loss function, $\mathcal{L}_{\text{total}}$. This loss is a weighted sum of the individual losses from each of the four tasks.

$$\mathcal{L}_{\text{total}} = \lambda_{\text{id}}\mathcal{L}_{\text{id}} + \lambda_{\text{traj}}\mathcal{L}_{\text{traj}} + \lambda_{\text{vel}}\mathcal{L}_{\text{vel}} + \lambda_{\text{press}}\mathcal{L}_{\text{press}} \quad (21)$$

where λ are hyper-parameters that balance the task contributions. The individual loss functions are: $\mathcal{L}_{\text{id}}$ (Categorical Cross-Entropy), and $\mathcal{L}_{\text{traj}}$, $\mathcal{L}_{\text{vel}}$, $\mathcal{L}_{\text{press}}$ (Mean Squared Error).

The systematic optimization and training logic for the proposed model are detailed in the following section. Table 1 presents Algorithm 1, illustrating the DynaSigNet end-to-end training procedure.

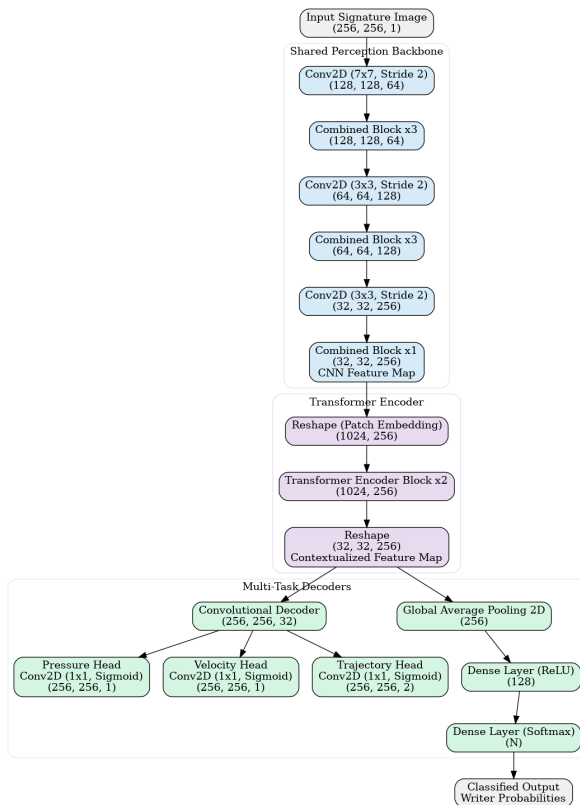


Fig. 2: The architecture of the DynaSigNet model. The model consists of a shared perception backbone and a set of multi-task decoders. The backbone uses a hybrid CNN-Transformer design to extract a Contextualized Feature Map. This map is then processed in parallel by an Identification Head for writer classification and by Motor-Imagination Heads to infer the latent dynamic properties (pressure, velocity, and trajectory) of the signature.

Source: created by the authors.

Table 1. DynaSigNet End-to-End Training Procedure (Algorithm 1).

```

Require: Training dataset  $\mathcal{D} = \{(\mathbf{X}^{(i)}, \hat{\mathbf{y}}_{\text{id}}^{(i)}, \hat{\mathbf{Y}}_{\text{traj}}^{(i)}, \hat{\mathbf{y}}_{\text{vel}}^{(i)}, \hat{\mathbf{y}}_{\text{press}}^{(i)})\}_{i=1}^M$ 
Require: Learning rate  $\alpha$ , balancing weights  $\{\lambda_{\text{id}}, \lambda_{\text{traj}}, \lambda_{\text{vel}}, \lambda_{\text{press}}\}$ 
1: Initialize: Model parameters  $\theta$ 
2: for each epoch do
3:   for each batch  $\{\mathbf{X}, \hat{\mathbf{y}}_{\text{id}}, \hat{\mathbf{Y}}_{\text{traj}}, \hat{\mathbf{y}}_{\text{vel}}, \hat{\mathbf{y}}_{\text{press}}\}$  in  $\mathcal{D}$  do
4:     1. Forward Pass (Backbone):
5:      $\mathbf{F}_{\text{cnn}} \leftarrow \text{CNN\_Extractor}(\mathbf{X}; \theta_{\text{cnn}})$ 
6:      $\mathbf{Z} \leftarrow \text{Transformer\_Encoder}(\mathbf{F}_{\text{cnn}}; \theta_{\text{trans}})$ 
7:     2. Forward Pass (Decoders):
8:      $\mathbf{y}_{\text{id}} \leftarrow \text{Identification\_Head}(\mathbf{Z}; \theta_{\text{id}})$ 
9:      $\mathbf{H}_{\text{dyn}} \leftarrow \text{Convolutional\_Decoder}(\mathbf{Z}; \theta_{\text{dec}})$ 
10:     $\mathbf{Y}_{\text{traj}} \leftarrow \text{Trajectory\_Head}(\mathbf{H}_{\text{dyn}}; \theta_{\text{traj}})$ 
11:     $\mathbf{y}_{\text{vel}} \leftarrow \text{Velocity\_Head}(\mathbf{H}_{\text{dyn}}; \theta_{\text{vel}})$ 
12:     $\mathbf{y}_{\text{press}} \leftarrow \text{Pressure\_Head}(\mathbf{H}_{\text{dyn}}; \theta_{\text{press}})$ 
13:    3. Loss Calculation:
14:     $\mathcal{L}_{\text{id}} \leftarrow \text{CrossEntropy}(\mathbf{y}_{\text{id}}, \hat{\mathbf{y}}_{\text{id}})$ 
15:     $\mathcal{L}_{\text{traj}} \leftarrow \text{MSE}(\mathbf{Y}_{\text{traj}}, \hat{\mathbf{Y}}_{\text{traj}})$ 
16:     $\mathcal{L}_{\text{vel}} \leftarrow \text{MSE}(\mathbf{y}_{\text{vel}}, \hat{\mathbf{y}}_{\text{vel}})$ 
17:     $\mathcal{L}_{\text{press}} \leftarrow \text{MSE}(\mathbf{y}_{\text{press}}, \hat{\mathbf{y}}_{\text{press}})$ 
18:    4. Composite Loss:
19:     $\mathcal{L}_{\text{total}} \leftarrow \lambda_{\text{id}}\mathcal{L}_{\text{id}} + \lambda_{\text{traj}}\mathcal{L}_{\text{traj}} + \lambda_{\text{vel}}\mathcal{L}_{\text{vel}} + \lambda_{\text{press}}\mathcal{L}_{\text{press}}$ 
20:    5. Backpropagation and Optimization:
21:    Compute gradients:  $\nabla_{\theta} \mathcal{L}_{\text{total}}$ 
22:    Update parameters:  $\theta \leftarrow \theta - \alpha \nabla_{\theta} \mathcal{L}_{\text{total}}$ 
23:   end for
24: end for
25: return Trained model parameters  $\theta$ 

```

Source: created by the authors.

4 Experiment

Experiments were performed on a computer system featuring 16GB of RAM, an Intel Core i5 processor, and a dedicated GPU to improve training efficiency. The experiments were executed on a Windows 10 Pro (version 20H2) environment, leveraging TensorFlow 2.0.0 with cuDNN for optimized computational performance. Detailed descriptions of the dataset and the training protocols are provided in the following sections.

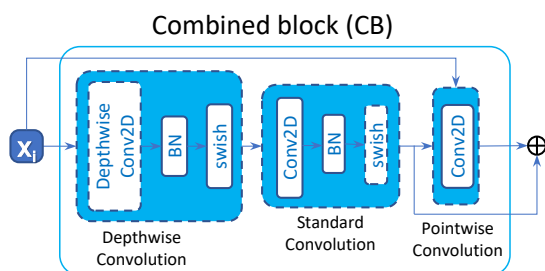
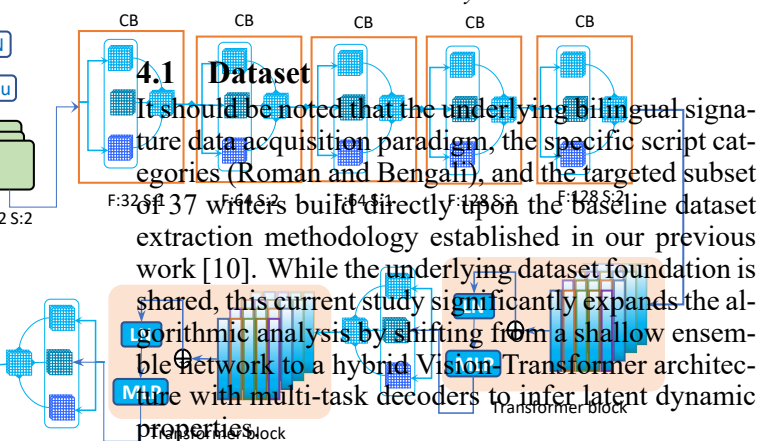


Fig. 3: The structure of the Combined Block (CB).

Source: created by the authors.



The signature-based writer identification dataset consists of contributions from 37 individual writers, encompassing a total of 3612 signature images. This dataset includes 1848 Roman script signatures and 1764 Bengali script signatures, reflecting a diverse mix of linguistic and stylistic variations. A selection of sample signatures from the dataset is illustrated in Figure 4, highlighting the diversity in writing styles across Roman and Bengali scripts. Such diversity makes the dataset a rich resource for developing and evaluating writer identification algorithms. It captures intricate handwriting characteristics, providing an ideal benchmark for both training and testing for our model.

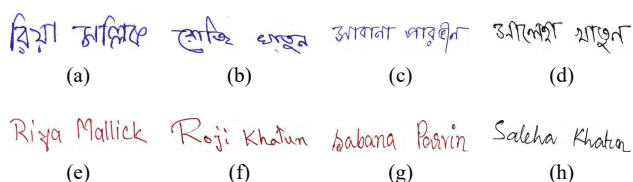


Fig. 4: Our dataset's sample images are organized as follows: Figures (a) and (d) in the first row show Bangla script signatures, while Figures (e) and (h) in the second row display Roman script signatures.

Source: [10].

The bilingual aspect allows for the exploration of cross-script influences, providing valuable insights

into how different writing systems affect handwriting. The extensive nature of the dataset ensures robust analysis, supporting applications in forensic investigations, banking security, and machine learning. To enhance computational efficiency, all images are converted to grayscale, focusing attention on the key signature features such as shape and texture while minimizing processing demands. The images are then resized to a standardized 300x100 pixel dimension, ensuring consistent input size for machine learning models. The dataset is meticulously labeled and divided into training and testing subsets, enabling effective model training and evaluation. In Figure 4 the sample signature images of the persons in Bangla and Roman are depicted.

It is important to note that the underlying bilingual signature data acquisition paradigm, including the specific script categories (Roman and Bengali) and the targeted subset of 37 writers, builds directly upon the baseline dataset extraction methodology established in our previous work [10]. While the foundational dataset is shared, the primary contribution of this current study lies in its algorithmic advancement. Specifically, this work transitions away from static, shallow ensemble networks, introducing a novel dynamic inference framework that utilizes a hybrid CNN-Transformer architecture and multi-task decoders to model the latent generative process of the signatures

4.2 Training regime

The DynaSigNet model was trained using both the Bangla script signatures and the Roman script signatures of 37 authors, each of whom contributed 52 signatures. The same 37 authors provided 52 signatures each in Bangla script, which were used for testing. For 100, 150, and 200 epochs, the model was trained with a batch size of 16 and a dropout rate of 0.5. The model's accuracy and efficiency throughout various training times could be assessed thanks to this step-wise training methodology. The suggested model's efficacy was then evaluated and documented.

5 Results

An extensive ablation study was performed to validate our architectural design, with results shown in Table 2. Each major component of the DynaSigNet model provides a clear, cumulative benefit. A baseline CNN-only model (M1) achieved 90.78% accuracy. Adding the Transformer encoder for global context (M2) raised this to 94.22%. Subsequently, incorporating the trajectory reconstruction task (M3) further increased performance to 95.89%. The final model (M4), our complete DynaSigNet with all

motor-imagination heads (pressure, velocity, and trajectory), achieved the highest accuracy of 98.103%. This demonstrates that each component local feature extraction, global context modeling, and latent dynamic inference plays a critical and synergistic role in achieving state-of-the-art performance in writer identification.

Table 3 presents the classification performance metrics for Bangla script recognition across different batch sizes (8, 16, and 32). Each class's performance is evaluated using precision, recall, and F1-score. With a batch size of 8, precision is generally high, often reaching 0.97 or 0.98, indicating good accuracy when the model does classify an instance into a specific class. However, recall varies more significantly, revealing differences in the model's ability to identify all instances of each class correctly. Notably, class 7 exhibits a lower recall (0.52), resulting in a reduced F1-score (0.68) with a batch size of 8. Furthermore, class 2 exhibits a similarly low recall of 0.59 and an F1-score of 0.73 at this batch size.

While a batch size of 16 yields consistently stronger performance across most classes, a batch size of 32 displays significant variability. For instance, class 5's recall drops to 0.42, leading to a low F1-score of 0.59. Crucially, class 12 also suffers an identical, massive drop to 0.44 recall under batch size 32. This erratic behavior across specific classes undermines the claim that a batch size of 16 is universally optimal merely due to minor smoothing. Instead, it indicates that certain writer profiles are highly sensitive to batch size scaling and gradient estimation variances.

Table 2. Comprehensive Ablation Study of DynaSigNet Architecture. Metrics show the cumulative contribution of each major module (CNN, Transformer, and Dynamic Heads) to overall writer identification accuracy.

Model version	CNN features	Transformer	Trajectory	Motor (P/V)	Accuracy (%)
M1: Baseline (CNN-Only)	✓	×	×	×	90.78
M2: Add Global Context	✓	✓	×	×	94.22
M3: Add Trajectory Guidance	✓	✓	✓	×	95.89
M4: Add Full Dynamic Heads	✓	✓	✓	✓	98.10

Source: created by the authors.

Table 3. Classification performance metrics of Bangla scripts with Batch size 8, 16, and 32.

#Class	Batch size 8			Batch size 16			Batch size 32		
	P	R	F1	P	R	F1	P	R	F1
0	0.98	0.92	0.97	0.99	0.97	0.98	0.65	0.98	0.78
1	0.98	0.97	0.97	0.98	0.97	0.97	1.00	0.96	0.98
2	0.97	0.59	0.73	0.97	0.96	0.96	0.96	0.90	0.93
3	0.78	0.97	0.86	0.98	0.97	0.97	0.90	0.98	0.94
4	0.98	1.00	0.99	0.99	0.98	0.98	0.98	0.95	0.96
5	0.99	0.98	0.98	0.98	0.99	0.98	0.97	0.42	0.59
6	0.98	0.88	0.93	0.97	0.98	0.97	0.97	0.90	0.93
7	0.98	0.52	0.68	0.98	0.98	0.98	0.97	0.92	0.94
8	0.76	0.98	0.86	0.99	0.97	0.98	0.91	0.99	0.95
9	0.97	0.94	0.95	0.97	1.00	0.98	0.98	0.95	0.96
10	0.97	0.99	0.98	0.97	0.98	0.97	0.94	0.96	0.95
11	0.98	0.98	0.98	0.98	0.99	0.98	0.98	1.00	0.99
12	0.99	0.94	0.96	0.98	0.98	0.98	0.98	0.44	0.61
13	0.95	0.98	0.96	0.98	0.97	0.97	0.62	0.97	0.76
14	0.94	0.94	0.94	0.99	0.97	0.98	0.99	0.96	0.97
15	0.99	1.00	0.99	0.96	0.98	0.97	0.96	0.92	0.94
16	0.99	0.98	0.98	0.98	0.96	0.97	0.91	0.97	0.94
17	0.98	0.97	0.97	0.99	0.95	0.97	0.95	0.97	0.96
18	0.99	0.90	0.94	0.96	0.94	0.95	0.97	0.94	0.95
19	0.98	0.99	0.98	0.97	0.95	0.96	0.81	0.96	0.88
20	0.98	0.94	0.96	0.97	0.95	0.96	0.99	0.98	0.98
21	0.98	0.99	0.98	0.98	0.97	0.97	0.98	0.96	0.97
22	0.98	0.94	0.96	0.97	0.97	0.97	0.98	0.97	0.97
23	0.90	0.99	0.94	0.96	0.96	0.96	0.96	0.98	0.97
24	0.98	0.98	0.98	0.98	0.97	0.97	0.93	0.98	0.95
25	0.72	0.99	0.83	0.99	0.96	0.97	0.97	0.99	0.98
26	0.98	0.95	0.96	0.99	0.96	0.97	0.98	0.83	0.90
27	0.79	0.97	0.87	0.99	0.96	0.97	0.95	0.99	0.97
28	0.99	0.98	0.98	0.97	0.98	0.97	0.93	0.96	0.94
29	0.98	0.96	0.97	0.97	0.97	0.97	0.96	0.97	0.96
30	0.97	0.98	0.97	0.96	0.96	0.96	0.98	0.94	0.96
31	0.92	0.67	0.78	0.98	0.98	0.98	0.98	0.98	0.98
32	0.80	0.98	0.88	0.96	0.98	0.97	0.96	0.92	0.94
33	0.99	0.98	0.98	1.00	0.97	0.98	0.97	0.85	0.91
34	0.98	0.98	0.98	0.97	0.96	0.96	0.99	0.83	0.90
35	0.99	0.98	0.98	0.97	0.96	0.96	0.85	0.99	0.91
36	0.96	0.99	0.97	0.96	0.98	0.97	0.91	0.98	0.94

P: precision; R: recall; F1: F1-score.

Source: created by the authors.

Table 4 details the classification performance metrics: Precision, Recall, and F1-score for Roman script recognition across three batch sizes: 8, 16, and 32. This analysis examines how varying batch sizes affect model accuracy and stability across different script classes (labeled 0-36).

Using a batch size of 8, the model demonstrates generally strong performance (e.g., class 0 achieves 0.95 precision, 0.93 recall, and 0.98 F1-score). However, there is a notable performance loss for class 10, where recall drops to 0.62, resulting in a significantly lower F1-score of 0.76. Shifting to a batch size of 16, we observe a performance gain for class 10, with precision, recall, and F1-score all rising to around 0.98. This suggests that a larger batch size helps the

model better learn the distinguishing features of this class. While most other classes maintain similar performance levels compared to batch size 8, there's a slight performance loss observed in classes 3, 6, 17, 18, and 34 with F1 scores of 0.89, 0.95, 0.93, 0.91, and 0.91, respectively.

Increasing the batch size to 32, the model achieves consistently high precision (around 0.97-0.98) and recall across nearly all classes. The performance gain is that the variability seen with smaller batch sizes is reduced, leading to more consistent performance. However, there's a small performance loss in recall for some classes such as class 6, 26, and 31, indicating that while the model remains accurate, it might be missing some instances due to the larger batch size potentially smoothing out finer class distinctions. While all batch sizes generally provide high accuracy, batch size 32 offers the most consistent and robust performance across classes, potentially due to better gradient estimation. However, the trade-off is a potential slight reduction in recall for a few classes. Batch size 16 improves performance on previously challenging classes but may introduce a small performance regression on other classes compared to 8 or 32.

The Table 5 shows the accuracies of a model for Bangla and Roman scripts across three batch sizes (8, 16, and 32). For the Bangla script, the accuracy is highest at a batch size of 16 (99%), but slightly drops to 94% with a batch size of 32. In contrast, the Roman script maintains a high accuracy of 98% for both batch sizes 8 and 16, and improves slightly to 99% at batch size 32. These results suggest that for Bangla, a batch size of 16 yields the best performance, while the Roman script benefits from larger batch sizes, with accuracy either remaining stable or slightly increasing.

Comparative study: The Table 6 in highlights the performance gains achieved by the proposed model over state-of-the-art architectures (VGG19, ResNet, and DenseNet121) for writer identification in Bangla and Roman signatures. The proposed model outperforms the best-performing baseline, ResNet, with a gain of 2.48% in precision (98.50% vs. 96.02%), 1.13% in recall (98.44% vs. 97.31%), 1.80% in F1-score (98.46% vs. 96.66%), and 1.52% in accuracy (98.50% vs. 96.98%). Compared to DenseNet121, the gains are even more significant, with an accuracy improvement of 4.38% (98.50% vs. 94.12%). These results underscore the superior ability of the proposed model to extract and utilize distinctive features, achieving substantial gains over standard architectures.

Table 4. Classification performance metrics of Roman scripts with Batch size 8, 16, and 32.

#Class	Batch size 8			Batch size 16			Batch size 32		
	P	R	F1	P	R	F1	P	R	F1
0	0.93	0.98	0.95	0.97	0.98	0.97	0.98	0.98	0.98
1	0.98	0.98	0.98	0.97	0.98	0.97	0.98	0.98	0.98
2	0.97	0.97	0.97	0.97	0.99	0.98	0.97	0.99	0.98
3	0.98	0.98	0.98	0.81	0.98	0.89	0.91	0.97	0.94
4	0.93	0.98	0.95	0.94	0.96	0.95	0.98	0.98	0.98
5	0.97	0.92	0.94	0.98	0.95	0.96	0.97	0.98	0.97
6	0.97	0.85	0.91	0.98	0.92	0.95	0.98	0.81	0.89
7	0.97	0.98	0.97	0.94	0.98	0.96	0.95	0.98	0.96
8	0.96	0.98	0.97	0.96	0.98	0.97	0.98	0.98	0.98
9	0.98	0.95	0.96	0.98	0.99	0.98	0.98	0.97	0.97
10	0.97	0.62	0.76	0.98	0.98	0.98	0.95	0.98	0.96
11	0.98	0.94	0.98	0.95	0.98	0.96	0.97	0.97	0.97
12	0.99	0.98	0.98	0.98	0.98	0.98	0.95	0.98	0.96
13	0.96	0.95	0.95	0.97	0.98	0.97	0.98	0.97	0.97
14	0.91	0.98	0.94	0.98	0.98	0.98	0.98	0.98	0.98
15	0.98	0.98	0.98	0.91	0.97	0.94	0.97	0.98	0.97
16	0.93	0.98	0.95	0.98	0.98	0.98	0.96	0.98	0.97
17	0.99	0.97	0.98	0.99	0.88	0.93	0.97	0.98	0.97
18	0.98	0.96	0.97	0.98	0.85	0.91	0.96	0.98	0.97
19	0.97	0.99	0.98	0.97	0.96	0.96	0.97	0.98	0.97
20	0.93	0.96	0.94	0.91	0.96	0.93	0.96	0.98	0.97
21	0.98	0.95	0.96	0.98	0.96	0.97	0.96	0.98	0.97
22	0.96	0.99	0.97	0.97	0.96	0.96	0.98	0.95	0.96
23	0.98	0.98	0.98	0.96	0.99	0.97	0.99	0.98	0.98
24	0.98	0.87	0.92	0.97	0.99	0.98	0.96	0.98	0.97
25	0.96	0.98	0.97	0.98	0.96	0.97	0.98	0.96	0.97
26	0.98	0.94	0.96	0.98	0.98	0.98	0.99	0.92	0.95
27	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98
28	0.98	0.98	0.98	0.99	0.97	0.98	0.98	0.98	0.98
29	0.97	0.92	0.94	0.97	0.88	0.92	0.98	0.96	0.97
30	0.98	0.98	0.98	0.98	0.97	0.97	0.97	0.98	0.97
31	0.98	0.99	0.98	0.97	0.98	0.97	0.98	0.94	0.96
32	0.98	0.99	0.98	0.99	0.99	0.99	0.98	0.95	0.96
33	0.96	0.99	0.97	0.98	0.96	0.97	0.98	0.97	0.97
34	0.98	0.97	0.97	0.94	0.88	0.91	0.96	0.98	0.97
35	0.98	0.97	0.97	0.98	0.98	0.98	0.98	0.98	0.98
36	0.96	0.92	0.94	0.96	0.98	0.97	0.93	0.96	0.94

P: precision; R: recall; F1: F1-score.

Source: created by the authors.

Table 5. The accuracies of Bangla and Roman for different batch sizes.

Script	Batch size		
	8	16	32
Bangla	0.95	0.99	0.94
Roman	0.98	0.98	0.99

Source: created by the authors.

5.1 GUI Interface

The graphical user interface (GUI) of the Signature Verification System, shown in Figure 5 is built for intuitive and easy to access, employing a combination of PHP for backend functionality, HTML and CSS for front-end presentation, and Flask for API integration. The "Upload Signature" frame on the left side is written in HTML and styled with CSS to display a

clean and user-friendly interface for uploading signature images. Users can opt to drag-and-drop images into the target zone or click to open their local file system, made available by JavaScript for interactive use. PHP handles the file uploading, storing the image temporarily securely on the server. When the "Verify Signature" button (CSS also styled) is clicked, PHP makes an API call to a Flask backend, passing the image data for processing.

The Flask backend receives the image data, pre-processed it, and feeds it into our Hybrid Visual Transformer model (.h5 file) to compute the identity verification score and signature classification. Flask returns the results back to the PHP frontend. The "Verification Result" column, dynamically updated by PHP, displays the uploaded signature, the identified script (e.g., Bangla, Roman), and the name of the claimed person. The Flask backend acts as a critical interface between the front-end interface and the core deep learning model, carrying out all computationally intensive tasks and ensuring efficient communication between the different system components. This design establishes a separation of concerns, wherein the presentation is handled by HTML/CSS and PHP and complicated model processing handled by the Flask backend, and the end result is a responsive and manageable system.

Table 6. Comparison of the proposed model with the state-of-the-art models. All scores are measured in %.

Model	P	R	F1	Accuracy
VGG19 [31]	95.57	96.62	96.09	96.76
ResNet [32]	96.02	97.31	96.66	96.98
DenseNet121 [33]	93.28	94.19	93.73	94.12
Proposed	98.50	98.44	98.46	98.50

P: precision; R: recall; F1: F1-score.

Source: created by the authors.

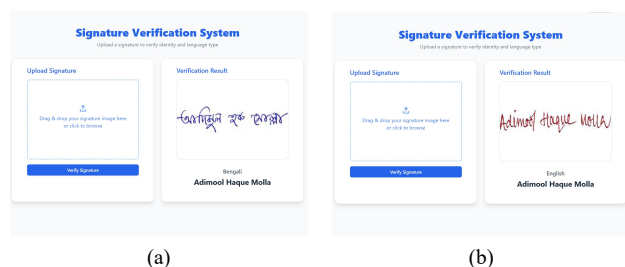


Fig. 5: The Signature Verification System accurately identifies the signature of the person.

Source: created by the authors.

6 Conclusions

In conclusion, this paper successfully introduced DynaSigNet, a novel deep learning framework that pioneers a new paradigm for offline writer identification. By moving beyond static pattern recognition, our model learns to infer the latent neuro-motor dynamics—such as signing trajectory, pressure, and velocity—from a single 2D image. The architecture synergistically combines a CNN-based backbone for fine-grained local feature extraction with a Transformer encoder for capturing global contextual relations. Our core innovation, a multi-task learning approach with auxiliary motor-imagination heads, compels the model to build a profoundly more discriminative feature representation. By achieving a state-of-the-art writer identification accuracy of 98.50% on a challenging bilingual dataset of Roman and Bengali scripts, this work validates our hypothesis that modeling the generative process is a highly effective strategy for script-invariant biometrics. The future scope of this research is significant. Follow-up efforts will focus on extending this dynamic inference framework to the more challenging task of expert-level forgery detection, where subtle variations in motor control are key differentiators. Furthermore, we plan to evaluate the model's resilience to real-world degradations and explore knowledge distillation techniques for deployment on low-end hardware in forensic and security applications. Ultimately, this work lays the foundation for a new class of biometric systems that bridge the gap between static and dynamic analysis.

References:

- [1] Reem Alrawili, Ali Abdullah S AlQahtani, and Muhammad Khurram Khan. Comprehensive survey: Biometric user authentication application, evaluation, and discussion. *Computers and Electrical Engineering*, 119:109485, 2024. <https://doi.org/10.1016/j.compeleceng.2024.109485>.
- [2] VV Starovoitov and U Yu Akhundjanov. A writer-dependent approach to offline signature verification based on one-class support vector machine. *Pattern Recognition and Image Analysis*, 34(2):340–351, 2024. <https://doi.org/10.1134/S1054661824700135>.
- [3] SD Bhavani and RK Bharathi. A multi-dimensional review on handwritten signature verification: strengths and gaps. *Multimedia Tools and Applications*, 83(1):2853–2894, 2024. <https://doi.org/10.1007/s11042-023-15357-2>.
- [4] Alexios Giazitzis, Moises Diaz, Elias N Zois, and Miguel A Ferrer. Janus-faced handwritten signature attack: A clash between a hand-

- written signature duplicator and a writer independent, metric meta-learning offline signature verifier. In *International Conference on Document Analysis and Recognition*, pages 216–232. Springer, 2024. https://doi.org/10.1007/978-3-031-70536-6_13.
- [5] Mridul Ghosh, Sayan Saha Roy, Himadri Mukherjee, Sk Md Obaidullah, KC Santosh, and Kaushik Roy. Understanding movie poster: transfer-deep learning approach for graphic-rich text recognition. *The Visual Computer*, 38:1645–1664, 2022. <https://doi.org/10.1007/s00371-021-02094-6>.
 - [6] José Mauricio, Inês Domingues, and Jorge Bernardino. Comparing vision transformers and convolutional neural networks for image classification: A literature review. *Applied Sciences*, 13(9):5521, 2023. <https://doi.org/10.3390/app13095521>.
 - [7] Mridul Ghosh, Sayan Saha Roy, Himadri Mukherjee, Sk Md Obaidullah, Xiao-Zhi Gao, and Kaushik Roy. Movie title extraction and script separation using shallow convolution neural network. *IEEE Access*, 9:125184–125201, 2021. <https://doi.org/10.1109/ACCESS.2021.3110858>.
 - [8] Liyang Xie, Zhongcheng Wu, Xian Zhang, Yong Li, and Xinkuang Wang. Writer-independent online signature verification based on 2d representation of time series data using triplet supervised network. *Measurement*, 197:111312, 2022. <https://doi.org/10.1016/j.measurement.2022.111312>.
 - [9] Victor KSL Melo, Byron Leite Dantas Bezerra, Donato Impedovo, Giuseppe Pirlo, and Antonio Lundgren. Deep learning approach to generate offline handwritten signatures based on online samples. *IET Biometrics*, 8:215–220, 2019. <https://doi.org/10.1049/iet-bmt.2018.5091>.
 - [10] Sk Md Obaidullah, Mridul Ghosh, Himadri Mukherjee, Kaushik Roy, and Umapada Pal. Sen: Stack ensemble shallow convolution neural network for signature-based writer identification. In *2022 26th International Conference on Pattern Recognition (ICPR)*, pages 1414–1420. IEEE, 2022. <https://doi.org/10.1109/ICPR56361.2022.9956456>.
 - [11] Emre Özgündüz, Tülin Şentürk, and M. Elif Karslıgil. Off-line signature verification and recognition by Support Vector Machine. In *2005 13th European Signal Processing Conference (EUSIPCO)*, pages 1–4. IEEE, 2005. Accessed: 08 May 2026. <https://ieeexplore.ieee.org/abstract/document/7078479>.
 - [12] Muhammad Reza Pourshahabi, Mo-hamad Hoseyn Sigari, and Hamid Reza Pourreza. Offline handwritten signature identification and verification using contourlet transform. In *2009 International Conference of Soft Computing and Pattern Recognition*, pages 670–673. IEEE, 2009. <https://doi.org/10.1109/SoCPaR.2009.132>.
 - [13] Elaheh Soleymanpour, Boshra Rajae, and Hamid Reza Pourreza. Offline handwritten signature identification and verification using contourlet transform and support vector machine. In *2010 6th Iranian Conference on Machine Vision and Image Processing*, pages 1–6. IEEE, 2010. <https://doi.org/10.1109/IranianMVIP.2010.5941179>.
 - [14] Nabih Azizi, Nadir Farah, and Mokhtar Selami. Ensemble classifier construction for arabic handwritten recognition. In *International Workshop on Systems, Signal Processing and their Applications, WOSSPA*, pages 271–274. IEEE, 2011. <https://doi.org/10.1109/WOSSPA.2011.5931470>.
 - [15] Srikanta Pal, Alaei Alireza, Umapada Pal, and Michael Blumenstein. Off-line signature identification using background and foreground information. In *2011 International Conference on Digital Image Computing: Techniques and Applications*, pages 672–677. IEEE, 2011. <https://doi.org/10.1109/DICTA.2011.119>.
 - [16] Srikanta Pal, Umapada Pal, and Michael Blumenstein. Hindi and english off-line signature identification and verification. In *Proceedings of International Conference on Advances in Computing*, pages 905–910. Springer, 2012. https://doi.org/10.1007/978-81-322-0740-5_109.
 - [17] Srikanta Pal, Sukalpa Chanda, Umapada Pal, Katrin Franke, and Michael Blumenstein. Off-line signature verification using g-surf. In *2012 12th international conference on intelligent systems design and applications (ISDA)*, pages 586–591. IEEE, 2012. <https://doi.org/10.1109/ISDA.2012.6416603>.
 - [18] Srikanta Pal, Umapada Pal, and Michael Blumenstein. A two-stage approach for english and hindi off-line signature verification. In *New Trends in Image Analysis and Processing—ICIAP 2013: ICIAP 2013 International Workshops, Naples, Italy, September 9-13, 2013. Proceedings 17*, pages 140–148. Springer, 2013. https://doi.org/10.1007/978-3-642-41190-8_16.
 - [19] Pawan Kumar Singh, Ram Sarkar, Mita Nasipuri, and David Doermann. Word-level script identification for handwritten in-

- dic scripts. In *2015 13th International Conference on Document Analysis and Recognition (ICDAR)*, pages 1106–1110. IEEE, 2015. <https://doi.org/10.1109/ICDAR.2015.7333932>.
- [20] Durjoy Sen Maitra, Ujjwal Bhattacharya, and Swapan K Parui. Cnn based common approach to handwritten character recognition of multiple scripts. In *2015 13th International Conference on Document Analysis and Recognition (ICDAR)*, pages 1021–1025. IEEE, 2015. <https://doi.org/10.1109/ICDAR.2015.7333916>.
- [21] Alireza Alaei, Srikanta Pal, Umapada Pal, and Michael Blumenstein. An efficient signature verification method based on an interval symbolic representation and a fuzzy similarity measure. *IEEE Transactions on Information Forensics and Security*, 12(10):2360–2372, 2017. <https://doi.org/10.1109/TIFS.2017.2707332>.
- [22] Luiz G Hafemann, Robert Sabourin, and Luiz S Oliveira. Learning features for offline handwritten signature verification using deep convolutional neural networks. *Pattern Recognition*, 70:163–176, 2017. <https://doi.org/10.1016/j.patcog.2017.05.012>.
- [23] Priteshkumar Gohil, Santosh Thoduka, and Paul G Plöger. Sensor fusion and multimodal learning for robotic grasp verification using neural networks. In *2022 26th International Conference on Pattern Recognition (ICPR)*, pages 5111–5117. IEEE, 2022. <https://doi.org/10.1109/ICPR56361.2022.9955646>.
- [24] Tiong-Sik Ng, Cheng-Yaw Low, Jacky Chen Long Chai, and Andrew Beng Jin Teoh. Conditional multimodal biometrics embedding learning for periocular and face in the wild. In *2022 26th International Conference on Pattern Recognition (ICPR)*, pages 812–818. IEEE, 2022. <https://doi.org/10.1109/ICPR56361.2022.9956636>.
- [25] Matthias Körschens, Paul Bodesheim, and Joachim Denzler. Occlusion-robustness of convolutional neural networks via inverted cutout. In *2022 26th International Conference on Pattern Recognition (ICPR)*, pages 2829–2835. IEEE, 2022. <https://doi.org/10.1109/ICPR56361.2022.9956044>.
- [26] Marco Peer, Florian Kleber, and Robert Sablatnig. Writer retrieval using compact convolutional transformers and netmvlad. In *2022 26th International Conference on Pattern Recognition (ICPR)*, pages 1571–1578. IEEE, 2022. <https://doi.org/10.1109/ICPR56361.2022.9956155>.
- [27] Nicole Dalia Cilia, Tiziana D’Alessandro, Claudio De Stefano, and Francesco Fontanella. Offline handwriting image analysis to predict alzheimer’s disease via deep learning. In *2022 26th International Conference on Pattern Recognition (ICPR)*, pages 2807–2813. IEEE, 2022. <https://doi.org/10.1109/ICPR56361.2022.9956359>.
- [28] Younes Akbari, Noor Almaadeed, Somaya Al-Maadeed, Fouad Khelifi, and Ahmed Bouridane. Prnu-net: A deep learning approach for source camera model identification based on videos taken with smartphone. In *2022 26th international conference on pattern recognition (ICPR)*, pages 599–605. IEEE, 2022. <https://doi.org/10.1109/ICPR56361.2022.9956272>.
- [29] Mayank Kumar Kundalwal, Anurag Saraswat, Ishan Mishra, and Deepak Mishra. Baf: Federated learning with base ablation for cost effective communication. In *2022 26th International Conference on Pattern Recognition (ICPR)*, pages 1922–1928. IEEE, 2022. <https://doi.org/10.1109/ICPR56361.2022.9956684>.
- [30] Daniel Yapi and Mohand Said Allili. Multi-band texture modeling using finite mixtures of multivariate generalized gaussian distributions. In *2022 26th International Conference on Pattern Recognition (ICPR)*, pages 464–469. IEEE, 2022. <https://doi.org/10.1109/ICPR56361.2022.9956448>.
- [31] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. <https://doi.org/10.48550/arXiv.1409.1556>.
- [32] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. Accessed: 08 May 2026. https://www.cv-foundation.org/openaccess/content_cvpr_2016/html/He_Deep_Residual_Learning_CVPR_2016_paper.html.
- [33] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2261–2269, 2017. <https://doi.org/10.1109/CVPR.2017.243>.

Contribution of Individual Authors to the Creation of a Scientific Article (Ghostwriting Policy)

The authors contributed in the present research, at all stages from the formulation of the problem to the final findings and solution.

Sources of Funding for Research Presented in a Scientific Article or Scientific Article Itself

No funding was received for conducting this study.

Conflict of Interest

The authors have no conflicts of interest to declare that are relevant to the content of this article.

Creative Commons Attribution License 4.0 (Attribution 4.0 International, CC BY 4.0)

This article is published under the terms of the Creative Commons Attribution License 4.0

https://creativecommons.org/licenses/by/4.0/deed.en_US