Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

Advancing Optical Character Recognition for Low-Resource Scripts: A Siamese Meta-Learning Approach with PSN Framework

ANIRUDHA GHOSH[1], DEBADITYA BARMAN[1] **(Senior Member, IEEE)**, ABU SUFIAN[2,3] **(Member, IEEE)**, and IBRAHIM A. HAMEED [4, 5] **(Senior Member, IEEE)**

[1]Department of Computer and System Sciences, Visva-Bharati University, Santineketan, India. E-mail: 03333342106@visva-bharati.ac.in & E-mail: debadityabarman@gmail.com

[2]Institute of Applied Sciences and Intelligent Systems, Lecce, Italy & Department of Computer Science, University of Gour Banga, English Bazar, India. E-mail: abu.sufian@isasi.cnr.it

[3]Department of Computer Science, University of Gour Banga, English Bazar, India. E-mail: sufian@ugb.ac.in

[4]Department of ICT and Natural Sciences, Norwegian University of Science and Technology, Trondheim, Norway. E-mail: ibib@ntnu.no

[5]Department of Mechanical Engineering and Technology Management, Faculty of Science and Technology, Norwegian University of Life Sciences, Ås, Norway. E-mail: ibrahim.abdelhameed@nmbu.no

Corresponding author: Ibrahim A. Hameed (e-mail: ibib@ntnu.no, ibrahim.abdelhameed@nmbu.no).

ABSTRACT With the increasing demand for digitization, Optical Character Recognition (OCR) systems play a vital role in digitizing physical manuscripts. Several methods have been successfully deployed in the OCR domain. However, they often face challenges when dealing with low-resource regional scripts because of the limited training data and complex structure of characters. In such a scenario, Siamese Network (SN) meta-learning offers a promising solution for this problem by enabling quick adaptation to new tasks with minimal training data. Despite the success of SNs in various classification tasks, the traditional SN architecture seeks a compelling upgrade to improve its ability to distinguish between similar-looking characters of regional scripts. In this research paper, we propose a novel Priority-Smart Network (PSN) framework for traditional SN architectures, which can easily be incorporated into existing CNN backbone and improve their ability to identify characters in low-resource regional scripts. Furthermore, we propose the Enhanced Differential Edge Detection (EDED) preprocessing strategy explicitly designed for OCR tasks. With rigorous investigation and evaluation of three benchmark low-resource script datasets, we establish the effectiveness of our proposed techniques. Our experimentation results showcase significant advancements in character recognition accuracy and robustness, emphasizing the potential of SN combined with the PSN framework and EDED strategy for improving OCR systems in low-resource script.

INDEX TERMS Deep Learning, Low Resource Regional Languages, Meta-Learning, OCR, Priority-Smart Network, Siamese Network.

I. INTRODUCTION

IN the ever-evolving terrain of technology, Optical Character Recognition (OCR) plays a pivotal role in converting physical manuscripts into digital form [1]–[3]. With the increasing demand for digitization, there is an urgent need for OCR systems that can achieve superior accuracy and adaptability across various regional scripts, languages, and document formats. In India, 22 official languages and over 121 regional languages exist, most are considered low-resourced [4]. In this context, we refer to “low-resourced" which lacks adequate labeled data, linguistic models, and technological infrastructure [5], [6]. In such a scenario, OCR faces challenges, especially when handling the Low Resource Regional Languages (LR-RL) due to regional scripts’ inherent complexities and ambiguity [7].

Several methods have been successfully deployed in the OCR domain, ranging from earlier feature-matching techniques to more contemporary convolutional neural networks (CNN) based approaches [2], [6], [8]. In the context of LR-RL, these approaches encounter several challenges. Feature-matching-based techniques struggle with the complex fragments of these scripts, while current CNN-based approaches usually require vast amounts of training data. Furthermore, conventional OCR methods often face significant difficulties when dealing with LR-RL specailly Indian regional scripts, primarily due to the presence of characters that closely resemble each other and the intricate shapes of these characters. The absence of standard training datasets makes it even harder for regular OCRs to operate on these regional scripts. In this context, Siamese Meta Learning presents a promising solution, leveraging its proficiency in discerning similarities between entities. Moreover, the meta-learning aspect facilitates rapid adaptation to new tasks, addressing conventional OCR limitations in low-data scenarios and enhancing performance in low-resourced script recognition. However, despite the massive success of Siamese networks in various classification tasks [9]–[11], the traditional Siamese architecture seeks a compelling upgrade to improve its ability to distinguish between similar-looking characters of LR-RL. Conventional training methods of Siamese networks often lead to lagging performance and sub-optimal outcomes, extending concerns about their effectiveness.

To tackle the above-mentioned challenges, this work proposes a Priority-Smart Network (PSN) framework for Siamese networks. It's smart, lightweight, and can easily be incorporated into any existing CNN Siamese backbones, making them more efficient. The PSN framework works like an intelligent filter that pays more attention to important features. Moreover, it ensures a sophisticated and fine understanding of the underlying input data by granting adaptive priority to each extracted feature. By prioritizing and embedding features hierarchically from multiple levels of the model, our approach with PSN-incorporated models aims to tackle the challenges posed by high inter-class similarity among characters in LR-RL scripts.

The primary contributions through the outcomes of this investigation include:

1) A novel methodology leveraging the Siamese network-based meta-learning. It offers baseline accuracy in few-shot classification, setting the foundation across three LR-RL datasets: Olchiki, Assamese, and Meetei-Mayek (Mapi) Script. This contribution invites researchers to explore and advance this domain collaboratively.
2) We developed a novel PSN framework for few-shot character classification. Unlike traditional approaches using solely high-level features, PSN comprises hierarchical counts of features from multiple levels, improving the SN's discriminating power.
3) We propose an effective preprocessing technique- Enhanced Differential Edge Detection (EDED), fusing Improved Binary Patterns (IBP) with dynamic histogram enhancement and Canny edge detection. It uncovers and enhances the optical character's edges and hidden details.
4) We evaluated the performance of the proposed method extensively on the three LR-RL datasets. Experimental results shown that integrating PSN and EDED into the SN results in substantial performance gains with the state-of-the-art classical Siamese network.

The rest of the paper is organized as: Literature review in Section II, Materials and Method in Section III, Experimental details in Section IV, Results and Discussion in Section V, and Conclusion in Section VI.

II. LITERATURE REVIEW

Several studies emphasize challenges in utilizing traditional OCR for Low-Resourced Regional Languages (LR-RL) [2], [4], [7]. Early techniques, relying on template matching and feature extraction strategies, have laid the basis for OCR. Nevertheless, their efficacy decreases when faced with the diverse and complex nature of Indian scripts. High inter-class similarity, compounded by variability in font sizes and styles, has proven to be a substantial limitation for these traditional methods. This section discusses the related work across three intriguing aspects: The Evolution of the OCR system in Indian scripts, progress in OCR through Few-Shot Meta-learning techniques, and progress in Siamese Network Design for Few-Shot image classification.

A. THE EVOLUTION OF THE OCR SYSTEM IN LR-RL SCRIPTS

Recent literature explores the development of OCR systems customized explicitly for Indian scripts, handling the challenges of script intricacies and linguistic diversity [4]. This study briefly outlines the present character and numeral recognition landscape in offline handwritten Indian scripts, particularly for scripts like Assamese, Bangla, Malayalam, Oriya, Devanagari, Gurmukhi, and Tamil. Traditional OCR techniques include feature-based methods [12], template matching [13], statistical methods [14], neural networks [8], and hybrid approaches. Template matching applies by comparing predefined character templates against the input images. Feature-based approaches extract relevant features from input images, such as edges and textures; based on those extracted features, this strategy aims to classify any input image. Over the years, researchers developed OCR systems using several statistical methods, namely, Hidden Markov Models (HMM) and Conditional Random Fields (CRF), to develop OCR systems. In HMM-based OCR systems, characters are described as a series of visual features, and the model guesses the most probable sequence of characters based on the input image. CRF extends the concept of HMM by including contextual information to improve recognition accuracy. While neural networks, including Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN), input images are fed through layers containing convolutional filters to extract features, followed by dense layers for character recognition. RNN-based OCR techniques process images consecutively, treating each pixel column as a time step, allowing them to grab contextual information and reliances between two characters using recurrent relations. Hybrid approaches combine techniques like template matching and feature-based methods. Despite these strategies, challenges like variability in writing styles and limited training data persist, requiring further research for practical solutions.

Traditional OCR approaches often rely on large amounts of labeled data for training, which is scarce for low-resorce scripts. Few-shot Meta-learning stratiges can be used to solve this problem. This type of learning machanism stands out for its adaptability, delivering a slight and effective means of handling the challenges associated with low-resource regional scripts in OCR applications.

B. PROGRESS IN OCR THROUGH FEW-SHOT META-LEARNING TECHNIQUES

In recent years, a transformative shift has been observed in the field of development of OCR by integrating meta-learning techniques, especially emphasizing the power of few-shot learning (FSL) [15]–[17]. Meta-learning, often anointed with ‘‘learning to learn," promotes a model’s adaptability to novel tasks by disclosing it to diverse meta-tasks. Existing approaches fall into three types: metric-based (e.g., Matching Networks, Siamese Networks) [9], optimization-based (e.g., Model-Agnostic Meta-Learning (MAML)) [18], and model-based (e.g., Memory-Augmented Neural Network (MANN)) [19]. These methods facilitate quick learning on new tasks with the tiniest data, symbolizing a crucial paradigm evolution in machine learning for effective knowledge transfer. This section investigates the latest developments in meta-learning involved in OCR tasks, focusing on procedures that leverage FSL paradigms to improve the efficiency and adaptability of OCR systems.

[20] proposed Orc-Bert Augmenter, a novel strategy for few-shot oracle character recognition. They incorporate self-supervised learning into data augmentation, which addresses the challenges of imbalance and narrow data. Using a pre-trained self-supervised model, orc-Bert Augmentor generates augmented samples from unlabeled Chinese character datasets and a few labeled oracle characters. [21] employed double augmentation, applying standard transformations and a Conditional Deep Generative Adversarial Network (CGAN) for ancient Korean character recognition. They evaluate the results with the Lenet5 and Lenet8 network models. [22] suggested a statistical model for High Character Recognition, rendering multifarious image data based on character layout. [23] proposed a one-shot rule learning strategy for Malayalam Character Recognition, leveraging OneShot Hypothesis Derivation (OSHD), a logic schedule declarative bias for meta-level reasoning that focuses on high-level properties of the language. [24] propose an Adversarial Feature Learning (AFL) model for handwritten character recognition (HCR) that automatically comprises writer-independent semantic characteristics and uses standard printed data as objective primary knowledge. This approach achieves competitive results on MNIST and CASIA-HWDB datasets. [25] created a Siamese CNN for Tamil Handwritten Character Recognition acquiring optimal accuracy with tiniest shots. [26] applied a Siamese architecture for one-shot author verification using handwritten characters. They develop a novel CNN backbone for a Siamese network with three Convolutional layers and three fully connected layers; the model achieves an average verification accuracy on unseen test data. [27] proposed a Siamese network approach for handwritten Chinese character recognition (HCCR) using template matching to manage challenges such as fixed model size, raw data augmentation, and zero-shot learning abilities, showcasing its significance on ICDAR-2013 offline HCCR dataset.

While literature shows a narrow exploration of Indian scripts via few-shot meta-learning, a significant prospect exists for using meta-learning strategies in the low-resourced Indian script discipline. Notably, the extensively explored Siamese network stands out as a favorable solution for few-shot classification duties, offering a transformative strategy for handling the challenges associated with these scripts.

C. PROGRESS IN SIAMESE NETWORK DESIGN FOR FEW-SHOT IMAGE CLASSIFICATION

In the field of few-shot image classification, Siamese networks have grown immensely, addressing various challenges and stretching the limits of performance [28], [29]. Noteworthy works include [30] presenting a VGG-styled Siamese network for one-shot image classification and [31] offering a multi-resolution Siamese network. Contemporary innovations concentrate on attention mechanisms [[32], [33]], enhanced feature extraction [[34], [35]], dynamic routing [36], knowledge transfer [[37], [38]], memory-enhanced methods [39], and optimization strategies [40]. For example, the Cosine Siamese Network [41] acquired state-of-the-art performance utilizing cosine similarity, while the Capsule Siamese Network [42] presented capsule layers for pose-sensitive components. These improvements collectively contribute to developing Siamese network architectures, improving their effectiveness in few-shot image classification jobs.

The literature review illustrates the progress in Siamese networks in the field of few-shot image classification. These insights motivate us to integrate Siamese Meta-Learning with the proposed PSN framework to address the complicated challenges of Optical Character Recognition (OCR) in LR-RL scripts. By leveraging the improvement witnessed in Siamese network architectures, our approach expects to offer a targeted and valuable solution for improving OCR accuracy in LR-RL scripts, filling a vital gap in the present state of OCR technology by enabling context-aware feature extraction.

III. MATERIALS AND METHOD

A. FEW-SHOT LEARNING AND SIAMESE META-LEARNING

Few-shot learning (FSL) is a machine learning strategy that enables models to learn new tasks with very small amount of training data. It is also ideal for improving a model’s generalization and adaptation to new and unseen tasks.

While Siamese networks are specialized architecture experts at determining the likeness between instances, these models accomplish this by training on a support set holding

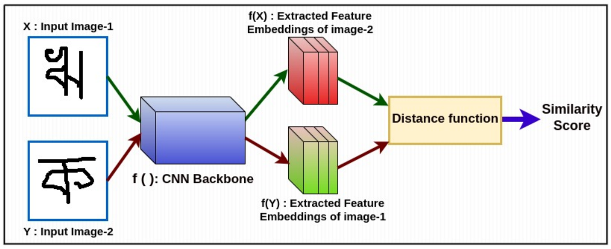
X : Input Image-1
Y : Input Image-2
f ( ): CNN Backbone
f(X) : Extracted Feature Embeddings of image-2
f(Y) : Extracted Feature Embeddings of image-1
Distance function
Similarity Score

FIGURE 1. Conceptual Architecture of Siamese Network.

only a few samples per class and a query set for testing or classification tasks. As illustrated in Figure 1, a Siamese network incorporates twin sub-networks, referred to as Siamese twins, serving as the network's backbone. This network takes a pair of images, denoted as (X, Y), and extracts feature embedding for each image, represented as $f(X)$ and $f(Y)$. The core function of the network involves using a distance calculation function to determine the similarity score between these input pairs. Siamese meta-learning combines Siamese networks with the meta-learning strategy to adapt to new tasks quickly. The model learns a universal similarity metric during meta-training, leveraging shared weights to capture commonalities across various tasks. This allows rapid adaptation to new tasks during the meta-testing phase, which is particularly practical for FSL in multiple domains.

While Siamese networks have demonstrated remarkable success in various tasks, a substantial structural challenge appears in the classical Siamese network architecture. This is particularly evident when classifying LR-RL utilizing a few-shot learning strategy in meta-learning techniques. These challenges are discussed below:

- **High script similarity:** LR-RL scripts show significant visual similarity, making it difficult to differentiate between them. Figure 2 illustrates two different character samples from the Gujarati script, showcasing remarkable visual similarity despite belonging to separate classes.

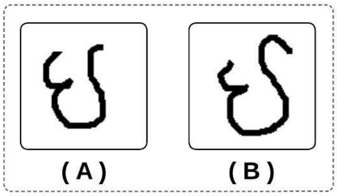

FIGURE 2. Two diffrent Gujarati characters with high visual similarity.

- **Scripts share features:** These scripts often share certain standard features that confuse the model and complicate the classification task.
- **Classification difficulty:** Siamese networks were initially crafted to compare similarities rather than standardize them into categories. To handle classification with Siamese networks effectively, it might be essential to integrate the networks with additional components.
- **Data handling:** Siamese networks generally require diverse good-quality training data for efficient LR-RL classification. Therefore, it is crucial to expose the network to various script styles and employ different data augmentation methods to enhance its performance.

Forming specific evaluation metrics that can capture the fine similarities between scripts is essential to tackle these challenges of the LR-RL classification task effectively. Our research concentrates on utilizing Siamese networks for OCR on LR-RL using three popular CNN architectures: ResNet [43], DenseNet [44], and EfficientNet [45]. By leveraging the capabilities of these architectures, we aim to attain good results in LR-RL, thereby advancing the field of low-resource script recognition.

B. FRAMEWORKS

We introduce the Priority-Smart Network (PSN) framework to enhance the performance of three CNN architectures for the Siamese network backbone: ResNet, DenseNet, and EfficientNet. We also systematically assess these architectures in two configurations: their original form and in partnership with the PSN framework. Our research offers a detailed comprehension of the significant impact of the PSN framework on performance metrics. Additionally, we introduce a preprocessing technique, named Enhanced Differential Edge Detection (EDED), specifically designed for Siamese networks in LR-RL character classification. EDED enhances edge detection to unveil subtle intricacies within textual data. The following two sections present a brief discussion on PSN and EDED techniques.

1) Priority-Smart Network (PSN) framework

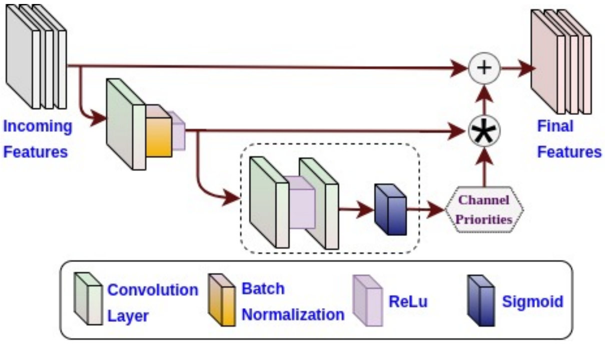

FIGURE 3. Architecture of PSN Framework.

The proposed Priority-Smart Network (PSN) framework can integrate into existing CNN frameworks. It is specifically designed to prioritize feature embeddings in OCR applications. Moreover, it can also act as an adaptive feature selector by dynamically highlighting important details. So, with the incorporation of the PSN framework into Siamese architectures, contextual attributes are emphasized, leading to context-aware feature extraction. Our approach integrates the PSN framework at multiple levels within the model architecture and hierarchically collects the feature embeddings for a rich and diverse feature representation. These embeddings play a crucial role in computing losses for the Siamese network, effectively mitigating challenges associated with high intra-class similarity in text classification tasks. This strategy ensures robust and accurate model performance, aligning with our objective of enhancing character recognition by preserving neighboring structures.

The building blocks PSN framework :

- **Convolutional Layer**: As shown in Figure 3, PSN framework begins with a 2D convolution operation, expressed as:

$$\text{Conv}_{\text{out}} = \text{Conv}_{3\times3}(\text{Input}) \tag{1}$$

where, $\text{Conv}_{3\times3}$ is a 3×3 convolution operation applied to the inputted feature maps. It aims to capture spatial information in the data.
- **Batch Normalization**: Then, the output is passed through batch normalization, which can be represented as:

$$\text{BatchNorm}_{\text{out}} = \frac{\text{Conv}_{\text{out}} - \mu}{\sqrt{\sigma^2 + \epsilon}} \tag{2}$$

where σ and μ are the standard deviation and mean of the output, and ϵ is a small constant used to prevent division by zero.
- **Rectified Linear Unit (ReLU)**: A ReLU activation function introduces non-linearity as:

$$\text{ReLU}_{\text{out}} = \max(0, \text{BatchNorm}_{\text{out}}) \tag{3}$$

It enhances the network's capability to capture complex patterns, and the complete feature flow is as:

$$\text{X} = \text{ReLU}(\text{BatchNorm}(\text{Conv}_{3\times3}(\text{Incoming\_Features}))) \tag{4}$$
- **Enhanced Attention Mechanism**: The core of the PSN framework involves an enhanced attention mechanism, incorporating two successive 1×1 convolutions and a ReLU activation. Here, the first convolution operation reduces the channel dimension by half, and then the second convolution operation restores it, aiming to learn spatial dependencies as:

$$\text{Attention}_{\text{out}} = \text{Conv}_{1\times1}(\text{ReLU}(\text{Conv}_{1\times1}(\text{X}))) \tag{5}$$

This mechanism allows the network to concentrate on specific feature channels, generating attention weights.
- **Sigmoid Activation**: To ensure that the attention weights remain within the range of 0 to 1, a sigmoid activation function is utilized as:

$$\text{Channel\_Weights} = \sigma(\text{Attention}_{\text{out}}) \tag{6}$$
- **Learnable Channel Priorities**: A key feature of the PSN framework is the introduction of learnable scalar priorities for each input feature channel. These priorities are represented as parameters (i.e. channel priorities), allowing the network to adaptively assign importance to different channels by scaling the attention weights with these channel priorities as:

$$\text{Channel\_modulated\_Weights} = \text{Channel\_Weights} \odot \text{Channel\_Priorities} \tag{7}$$
- **Feature Combination**: The resulting attention-modulated features (Channel_modulated_Weights) are element-wise multiplied (dot product) with the processed input (X), and the original input (Incoming_Features) is then added back to this product, promoting information flow and keeping the original features as:

$$\text{Final\_Features} = \text{Incoming\_Features} + (\text{X} \odot \text{Channel\_modulated\_Weights}) \tag{8}$$

This step ensures that crucial information is prioritized, making the PSN framework a strong tool for various applications.

2) Integration of PSN with Base Models

To tackle the challenge of OCR for LR-RL, we established the accuracy baseline using three proven Siamese network backbones: ResNet-18, DenseNet-169, and EfficientNet-B3. These CNN architectures are derived from the ImageNet contest. They have shown excellent recognition capability to handle a wide range of 1000 classes, delivering a strong foundation for our investigation.

To accommodate them into the complexities of LR-RL, we modified the last fully connected layer of their architecture from 1000 output nodes to 200. This architectural refinement is now recalled as ResNet-Structured Feature Extractor (RSFE), DensNet-Structured Feature Extractor (DSFE), and EfficientNet-Structured Feature Extractor (ESFE).

Extending this foundation, we integrated the PSN framework, a key element of our research, into the core of each of those refined architectures: RSFE-PSN, DSFE-PSN, and ESFE-PSN. This fusion was staged to evaluate the collective effective of PSN framework on LR-RL OCR tasks. The architectural details of each three investigated models are discussed below:

a: ResNet-Structured Feature Extractor (RSFE):

As shown in Figure 4, the initial architecture of our base model, named ResNet-Structured Feature Extractor (RSFE), follows the ResNet-18 framework and performs successively to extract features from the input data. It begins with a 7×7 convolutional layer followed by batch normalization (BN)

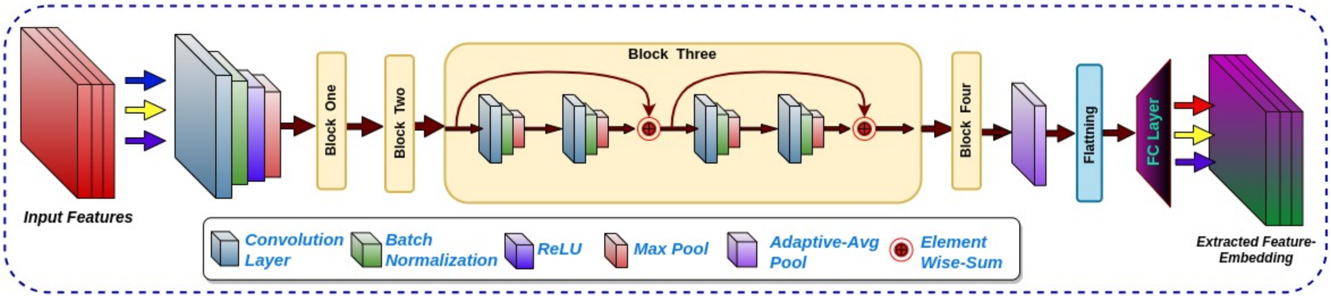

FIGURE 4. Architecture of the RSFE.

and a ReLU activation. Subsequently, spatial dimension reduction is achieved via max-pooling, symbolized as:

$$\begin{aligned}\text{Input} \rightarrow &Conv2d_{(7\times7)}(\,) \rightarrow \text{BN}(\,) \rightarrow \\ &\rightarrow \text{ReLU}() \rightarrow \text{MaxPool}(\,) \rightarrow\end{aligned} \tag{9}$$

The core of the architecture comprises four residual blocks: Block One, Block Two, Block Three, and Block Four. Each block has two sub-blocks with residual identity shortcuts: $Block_{ia}$ and $Block_{ib}$ and each sub-block is comprised of two consecutive 3×3 convolutions, followed by BN and ReLU activation, and the structure is expressed as:

$$\begin{aligned}&Block_{ia}\colon \{\text{Input(X)} \rightarrow Conv2d_{(3\times3)}(\,) \rightarrow \text{BN}() \rightarrow \\ &\quad\rightarrow \text{ReLU}() \rightarrow Conv2d_{(3\times3)}(\,) \rightarrow \text{BN}() \rightarrow \text{ReLU}()\} \\ &\rightarrow \text{Y}\colon \{Block_{ia} + \text{X}\} \rightarrow \\ &Block_{ib}\colon \{\text{Input(Y)} \rightarrow Conv2d_{(3\times3)}(\,) \rightarrow \text{BN}(\,) \rightarrow \\ &\quad\rightarrow \text{ReLU}() \rightarrow Conv2d_{(3\times3)}(\,) \rightarrow \text{BN}(\,) \rightarrow \text{ReLU}()\} \\ &\rightarrow \text{Z}\colon \{Block_{ib} + \text{Y}\} \rightarrow\end{aligned} \tag{10}$$

After feature extraction, an adaptive average pooling layer decreases spatial dimensions to 1×1. The model ends with a fully connected layer (FC) for classification, symbolized as:

$$\begin{aligned}\rightarrow \text{AvgPool}\colon &AdaptiveAvgPool2d_{(1\times1)}(\,) \rightarrow \\ &\rightarrow \text{FC Layer} \rightarrow \text{Output}\end{aligned} \tag{11}$$

This sequential arrangement allows the architecture to capture abstract and discriminatory features efficiently.

b: *RSFE incorporating PSN (RSFE-PSN) :*

As shown in Figure 5, we integrate the PSN framework followed by an average pooling layer and a fully connected layer after each of the four Blocks (Block-1, Block-2, Black-3, and Block4) of RSFE, Which helps prioritize deep features and extract complex hierarchical features.

c: *DensNet-Structured Feature Extractor (DSFE):*

Our second base model architecture, named the DensNet-Structured Feature Extractor (DSFE), follows the DensNet-169 framework. Figure 6 illustrates the DSFE. DSFE commences with a 7×7 initial Convolutional Layer with 64 output channels. This layer is augmented by Batch Normalization (BN) and ReLU activation function, forming the core structure:

$$\text{Conv}(7 \times 7, 64) \rightarrow \text{BN} \rightarrow \text{ReLU} \tag{12}$$

Next, we place four Dense Blocks in DSFE. Each Dense Block containins an ensemble of six Bottleneck Blocks (BB). Each Bottleneck Block sequence includes Batch Normalization (BN), ReLU activation, 1 × 1 convolution, optional dropout with rate d, Sigmoid activation, 3 × 3 convolution, and a concatenation operation at the end, symbolized as:

$$\text{DB}_1 : [\text{Bottleneck Block (BB)}_n]^6 \tag{13}$$

Where each BB can be represented as:

$$\begin{aligned}\text{BB}_n :&[\text{BN} \rightarrow \text{ReLU} \rightarrow \text{Conv}(1\times1) \rightarrow \{\text{Dropout}^d\} \rightarrow \\ &\rightarrow \text{BN} \rightarrow \sigma \rightarrow \text{Conv}(3\times3) \rightarrow \{\text{Dropout}^d\}] \rightarrow \\ &\rightarrow \text{Concatenate}\end{aligned} \tag{14}$$

The Transition Blocks follow the first three Dense Blocks, handling spatial dimensions through Conv(1 × 1) operations and Average Pooling as:

$$\begin{aligned}\text{Transition Block}_n : &[\text{BN} \rightarrow \text{ReLU} \rightarrow \text{Conv}(1\times1) \rightarrow \\ &\rightarrow \{\text{Dropout}^d\} \rightarrow \text{Avg Pool}(2\times2)]\end{aligned} \tag{15}$$

The DSFE model architecture is finalized with Batch Normalization (BN), ReLU activation, Global Average Pooling, and a Fully Connected (FC) Layer, represented as:

$$\begin{aligned}&\text{Input Features} \rightarrow \text{DB}_1 \rightarrow \text{Transition Block}_1 \rightarrow \\ &\rightarrow \text{DB}_2\colon [\text{Bottleneck}_n]^{12} \rightarrow \text{Transition Block}_2 \rightarrow \\ &\rightarrow \text{DB}_3\colon [\text{Bottleneck}_n]^{32} \rightarrow \text{Transition Block}_3 \rightarrow \\ &\rightarrow \text{DB}_4\colon [\text{Bottleneck}_n]^{32} \rightarrow \text{BN} \rightarrow \text{ReLU} \rightarrow \\ &\rightarrow \text{Global Average Pooling} \rightarrow \text{FC Layer}\end{aligned} \tag{16}$$

d: *DSFE incorporating PSN (DSFE-PSN):*

As presented in Figure 7, we integrate the PSN framework followed by an average pooling layer and a fully connected layer after each of the three Transition Blocks of DSFE. Additionally, another PSN framework is added after the last Dense Block(DB_3) of DSFE, and this PSN framework is also followed by an average pooling and a fully connected layer.

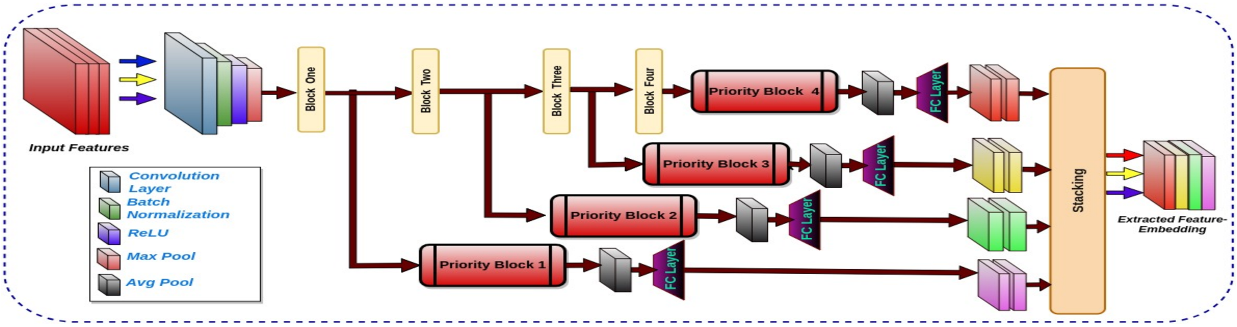

FIGURE 5. Architecture of the RSFE-PSN.

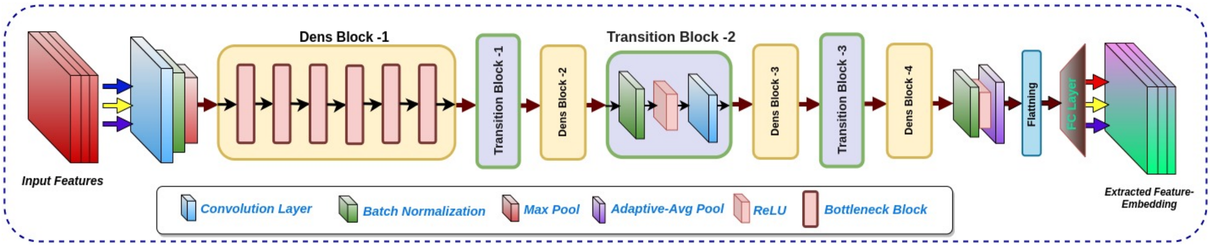

FIGURE 6. Architecture of the DSFE.

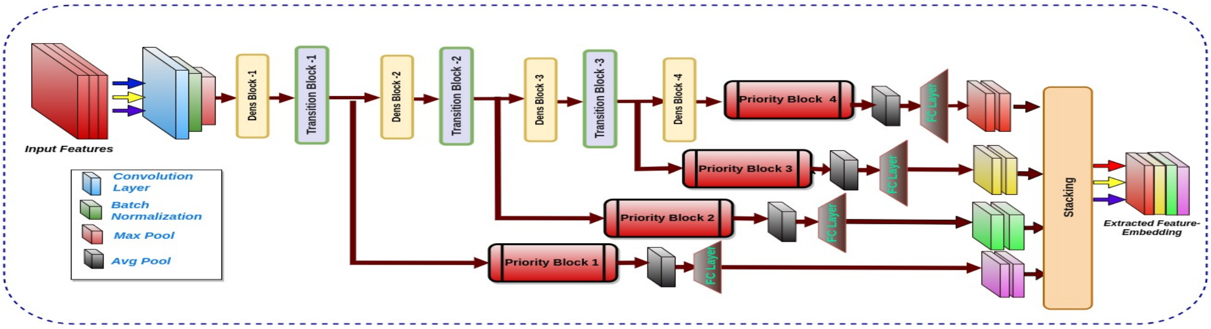

FIGURE 7. Architecture of the DSFE-PSN.

e: EfficientNet-Structured Feature Extractor (ESFE):

The third architecture of our base model, named the EfficientNet-Structured Feature Extractor (ESFE), adheres to the EfficientNet-B3 framework. Figure 8 presents the ESFE. ESFE begins with a 3 × 3 Convolutional Layer, followed by Batch Normalization (BN) and the SiLU (Sigmoid Linear Unit (σ)) activation function, represented as:

$$X_{\text{input}} \to \text{Conv}(3 \times 3) \to \text{BN} \to \sigma \tag{17}$$

The architecture of ESFE is segmented into distinct seven stages: stage 1 includes two Mobile Inverted Residual Bottleneck Convolution (MBConv) blocks with 1 × 1 convolution, both stage 2 and stage 3 encompasses three MBConv blocks with 6×6 convolution, stages 4 and 5 consist of five MBConv blocks with 6×6 convolution, stage 6 consists of six MBConv blocks with 6 × 6 convolution and stage 7 contains two MBConv6 blocks with 6 × 6 convolution.

Structure of MBConv Block: Each MBConv block exhibits a similar structure, comprising the following sub-blocks:

- **Expand Sub-Block**: This sub-block incorporates a 1×1 Convolution layer followed by batch normalization (BN) and the SiLU (σ) activation function as:

$$X_{\text{exp}} = \sigma(BN(\text{Conv2d}(X_{\text{in}}, C_{\text{in}}, 1, 1))) \tag{18}$$

- **Depthwise Sub-Block**: This sub-block utilizes a depthwise separable convolution layer (DW), which is followed by batch normalization and SiLU activation as:

$$X_{\text{dw}} = \sigma(BN(DW(X_{\text{exp}}, k, k))) \tag{19}$$

- **Squeeze-and-Excitation (SE) Sub-Block**: This sub-block facilitates channel-wise recalibration by employ-

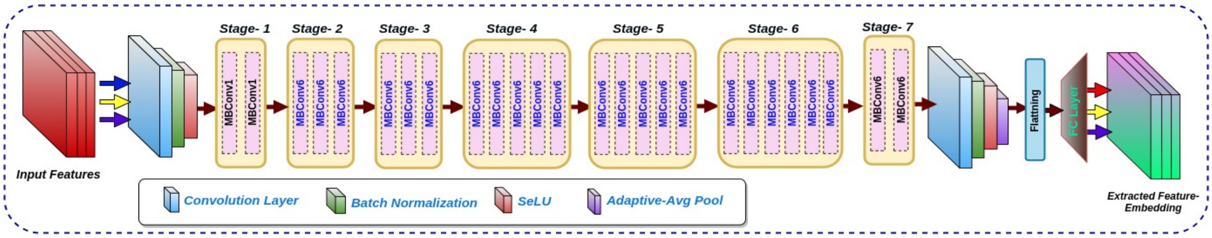

FIGURE 8. Architecture of the ESFE.

ing an adaptive average pooling layer (*AvgPool*) for dimension reduction as:

$$X_{\text{se}} = \text{AvgPool}(X_{\text{dw}}) \tag{20}$$

- A recalibration is further executed as:

$$X_r = \sigma(\text{Conv2d}(\sigma(\text{Conv2d}(X_{\text{se}}, \theta, 1, 1)), \sigma(\text{Conv2d}(X_{\text{se}}, \phi, 1, 1)))) \tag{21}$$

- **Reduction Sub-Block**: Defined as a 1×1 Convolution layer followed by batch normalization, this sub-block utilizes an identity activation function as:

$$X_{\text{reduced}} = BN(\text{Conv2d}(X_r, C_{\text{out}}, 1, 1)) \tag{22}$$

- A **DropSample** operation is integrated to enhance information flow and training efficiency.

After stage 7, we place a 1×1 Convolution operation followed by BN and the SiLU activation function:

$$X_{\text{head}} = \sigma(BN(\text{Conv2d}(X_{\text{last\_stage}}, C, k, k))) \tag{23}$$

To conclude the ESFE architecture, an adaptive average pooling denoted as *AvgPool*, followed by flattening and a linear (fully connected) layer with N output units is placed:

$$X_{\text{output}} = \text{Linear}(\text{Flatten}(\text{AvgPool}(X_{\text{head}})), N) \tag{24}$$

f: ESFE incorporating PSN (ESFE-PSN):

In Figure 9, we incorporate the PSN framework followed by an average pooling layer and a fully connected layer after each stage two, stage four, and stage six. Another PSN framework is added after the last stage of ESFE, and this PSN framework is also followed by an average pooling and a fully connected layer.

3) Enhanced Differential Edge Detection (EDED)

We propose an image preprocessing technique termed Enhanced Differential Edge Detection (EDED). EDED integrates Improved Binary Patterns (IBP), Dynamic Histogram Enhancement, and the well-established Canny edge detection approach to enhance the detection and highlighting of edges in images, amplifying minute elements and strengthening the overall feature quality within the character sample. By incorporating IBP, EDED captures and represents local texture information in image, while dynamic histogram enhancement facilitates adaptive contrast refinement as well as local contrast enhancement. Additionally, the inclusion of Canny edge detection improves precision in edge extraction. This fusion within EDED significantly augments image preprocessing and feature extraction processes, thereby substantially enhancing the capabilities of Optical Character Recognition (OCR) systems in character recognition and extraction of relevant information from images. The step-by-step algorithmic process of EDED is illustrated in Algorithm 1.

Algorithm 1: Abstract step-by-step Process of Enhanced Differential Edge Detection (EDED)

```
Data: image_path: Image path, radius:
      Neighborhood radius, threshold: Threshold,
      block_size: Histogram block size
Result: edge-enhanced image
Step 1: Initialization;
Load, grayscale, and resize image ($100 \times 100$),
 Initialize ibp_image as an empty 2D array;
Step 2: IBP Calculation;
for $EachPixel \in ResizedImage$ do
    Compute binary pattern: BinaryPattern($p$) =
      $\begin{cases} 1, & \text{if } |p - p_c| \geq \texttt{threshold} \\ 0, & \text{otherwise} \end{cases}$;
    Equalize histogram:
     EqualizedHistogram($H$) =
     HistogramEqualization($H$,block_size);
    Calculate mean value: MeanValue =
     $\frac{1}{N}\sum_{i=1}^{N} i \cdot$ EqualizedHistogram($i$);
    Weigh binary pattern to obtain IBP value:
     IBPValue($p$) =
     $\sum_{i=1}^{8}$ BinaryPattern($p_i$) $\cdot 2^i$;
    Store IBP value in ibp_image;
Step 3: Canny Edge Detection;
Apply Canny edge detection:
 Edges = Canny(ibp_image, $\sigma = 1$)
Step 4: Output;
Return edge-enhanced image;
```

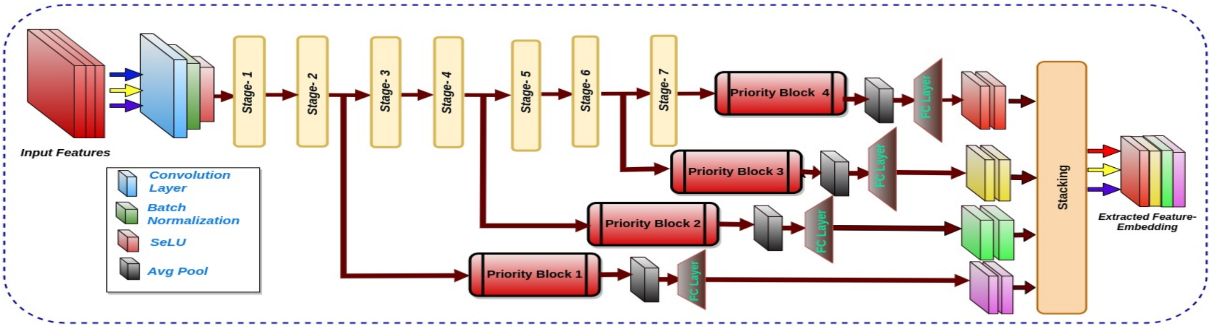

FIGURE 9. Architecture of ESFE-PSN.

4) Description of the Datasets

Our investigation employed multiple branch-mark datasets to train and evaluate the proposed OCR system to ensure its robustness and applicability.

a: Training Dataset

We use the Omniglot dataset to train all the models in our investigation. The Omniglot dataset is specially created for few-shot meta-learning or meta-training purposes, offering our models vast and varied examples to learn data insights. The structural components include:

- **Classes:** A comprehensive collection of 1623 characters spanning 50 distinct alphabets, collected form unique contact of multiple contributors via Amazon's Mechanical Turk. Each of the 1623 characters has 20 distinct samples.

b: Testing Dataset

To evaluate our strategy, we used three benchmark LR-RL OCR datasets, each depicting a unique linguistic context:

1) **Assamese Script:** A dataset of online handwritten Assamese characters, collaboratively gathered by 45 writers.
   - **Classes:** A vast collection comprising 183 classes containing 52 primary characters, 121 conjunct consonants, and 10 Assamese numerals.
2) **Manipuri Meetei-Mayek Script:** This dataset contains more than 5000 handwritten Manipuri Meetei-Mayek character samples across mixed demographics in Manipur. It includes characters from Cheitap Mayek, Cheising Mayek, Mapi Mayek, Lonsum Mayek, And Khutam Mayek.
   - **Classes:** A vast collection comprising 60 classes.
3) **OlChiki Script:** OlChiki, one of India's Scheduled languages, was formed in 1940 by Pandit Raghunath Murmu.
   - **Classes:** A vast collection comprising 30 characters, 24 consonants, and six vowels.

IV. EXPERIMENTS

In this section, we discuss the complete training process of Siamese networks of this investigation, leveraging state-of-the-art CNN-based RSFE, DSFE, and ESFE as the fundamental building blocks, our investigation extends across three phases: Base Model, Base Model with EDED, and Base Model Incorporated with PSN framework and EDED.

A. PREPROCESSING AND DATA AUGMENTATION:

Like traditional deep-learning approaches, efficient preprocessing is also critical in meta-learning. Our investigation with the Siamese network includes the following standard preprocessing strategy during the training of models:

- Resizing images to a uniform 100×100 resolution.
- Converting them to grayscale.
- Inverting colors for pattern accentuation.
- Transforming images into PyTorch tensors.

Further, we introduce Enhanced Differential Edge Detection (EDED), specially designed to improve the discrimination capabilities of the Siamese networks for OCR. To avoid the overfitting and vanishing gradients, we use data augmentation techniques. Arbitrary rotation introduces variability in image orientation with ±10 degrees of random rotation. Arbitrary Resized Crop diversifies image samples by random cropping and resizing to 100×100 pixels, with a scale varying from 90. Color Jitter adds variety in color attributes via random transformations, including brightness adjustments (up to 30). These strategies virtually extend the training dataset, promoting diverse and exhaustive model training.

B. TRAINING:

We train Siamese networks with RSFE, DSFE, and ESFE as backbone models. These networks learn to differentiate between negative and positive pairs randomly created from the training dataset, preserving an approximately 50-50% ratio balance. Each epoch or episode includes 5000 pairs, delivering rich data for the network to refine its discrimination capabilities. Vital hyper-parameters have a 30% dropout rate for the regularization and the use of the Adam optimizer with a learning rate of 0.0045 for efficient gradient-based optimization. We utilize a learning rate scheduler with a step size of 50 epochs and a gamma of 0.1 to enhance adaptability during training. The training transits 400 epochs, with model selection based on acquiring minimal validation loss, assuring peak generalization and performance.

a: Phase 1: Base Model Training with Standard Preprocessing:

In the initial phase of our investigation, we trained RSFE, DSFE, and ESFE on the training dataset for 400 epochs. We utilized predefined standard preprocessing and augmentation approaches during training, allotting 80% of the alphabet for training and 20% for validation. The resultant training and validation losses were documented and visualized in Figure 10, with a red dot denoting the minimum validation loss. This course provides a concise understanding of the model's performance and the impact of preprocessing and augmentation.

b: Phase 2: Base Model Training with EDED Preprocessing:

In the second phase, we conducted end-to-end training and validation of RSFE, DSFE, and ESFE on the training dataset for 400 epochs. Notably, we replaced the standard preprocessing strategies with our proposed EDED preprocessing technique. The training set contained 80% of the alphabet (40), with the remaining 20% allocated for validation. We recorded and visualized training and validation losses for each epoch throughout the process. Figure 11 illustrates the outcomes, with a red dot denoting the minimum validation loss. This stage aims to evaluate the efficiency of EDED in comparison to standard preprocessing approaches.

c: Phase 3: Base Model Incorporated with PSN framework Training with EDED Preprocessing:

In the third phase, again we performed the 400-epoch training and validation of RSFE-PSN, DSFE-PSN, and ESFE-PSN models on the training dataset. The base model architectures were significantly enhanced by incorporating our proposed PSN framework. The EDED preprocessing approach was consistently utilized. The training set contained 80% of the alphabet, with the remaining 20% used for validation. Results are visualized in Figure 12, with the red dot indicating the epoch of minimum validation loss. This phase evaluates the combined impact of the EDED preprocessing technique and the PSN framework on the performance of the RSFE-PSN, DSFE-PSN, and ESFE-PSNmodels.

V. RESULTS AND DISCUSSION

Our evaluation utilizes a K-Way N-Shot testing protocol. This FSL protocol is designed to rigidly evaluate the impact of the proposed PSN framework and EDED preprocessing strategy with different Siamese network backbones in real-world LR-RL OCR scenarios to identify the unknown character classes with a few support samples. It allows us to precisely measure the model's performance in bearing the character class's complexities, improving the reliability of models as potent solutions for real-world OCR applications. The algorithm for our K-Way N-Shot testing protocol is illustrated in Algorithm 2.

```
Algorithm 2: K-way N-shot Classification Test
Data: Dataset $D$, Model $f$, Evaluation Metric $M$
Result: Evaluation score $M(Q)$
Step 1: Data Splitting;
Let Dataset $D$, consisting of samples in $(x_i, y_i)$, where
  $x_i$ denotes an input and $y_i$ denotes corresponding
  class label;
Splitting $D$ into two subsets: Support Set ($D_S$) and
  Query Set ($D_Q$);
$S = \{(x_i, y_i)|y_i \in C_k,\ \text{for } k = 1 \text{ to } K, i = 1 \text{ to } N\}$;
$D_Q = D - D_S$;
*Support set consist of $N$ examples from each of the
  $K$ classes
*Query set consist of the remaining examples for
  testing
Step 2: Evaluation Metric;
Select an evaluation metric $M$, where $M$ can be
  precision, recall, accuracy, F1-score or any other
  metric;
Step 3: Testing Loop;
for $q \in Q$ do
    Select a support set $D_{S_q}$, holding $N$ examples
    from each of the $K$ classes, as $D_{S_q} = \{(x_i, y_i)|y_i \in$
    $C_k,\ \text{for } k = 1 \text{ to } K, i = 1 \text{ to } N\}$;
    Construct a mini-classification task
    $T_q = (q, D_{S_q})$;
    Use the trained model $f$ to make prediction for
    query example $q$: $f(q, D_{S_q}) = \text{predicted\_class}$;
    Evaluate the prediction by comparing it with the
    actual class label for $q$: $\delta(f(q, D_{S_q}), \text{true\_class})$;
Step 4: Calculate Metrics;
After iterating over all query examples, calculate the
  selected evaluation metric $M$ for the complete query
  set $Q$;
$M(Q) = \frac{\sum_{q \in Q} \delta(f(q, D_{S_q}), \text{true\_class})}{|D_Q|}$, where $q \in D_Q$;
```

A. PERFORMANCE EVALUATION, METRICS AND ANALYSIS FOR TEST DATASET-1: ASSAMESE SCRIPT

This section evaluates the base models (RSFE, DSFE, ESFE) and their enriched variants leveraging EDED preprocessing and the PSN framework for Assamese script classification. Consistent performance improvements are observed across 5-way, 8-way, and 12-way classifications with different shot scenarios. Table 1 presents the results of our evaluation process, where each data point is marked from 10 independent testing assignments, and the table depicts the mean and standard deviation for resulting data points. The integration of

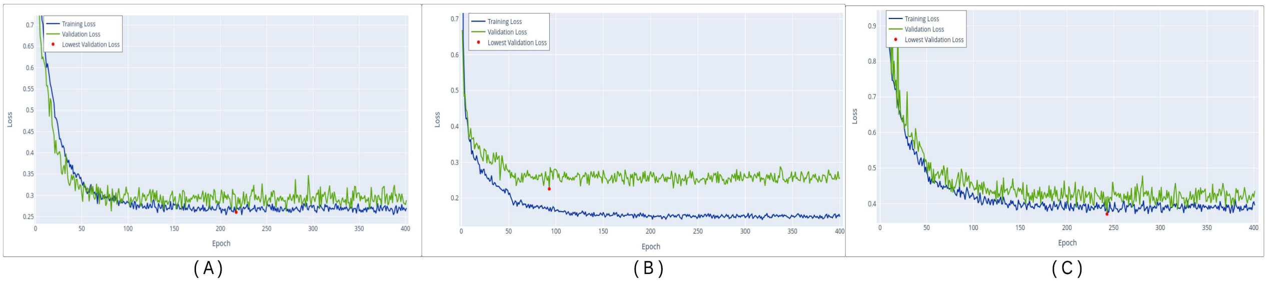

FIGURE 10. Number of epoch vs. training and validation loss (A. RSFE with standard preprocessing, B. DSFE with standard preprocessing and C. ESFE with standard preprocessing).

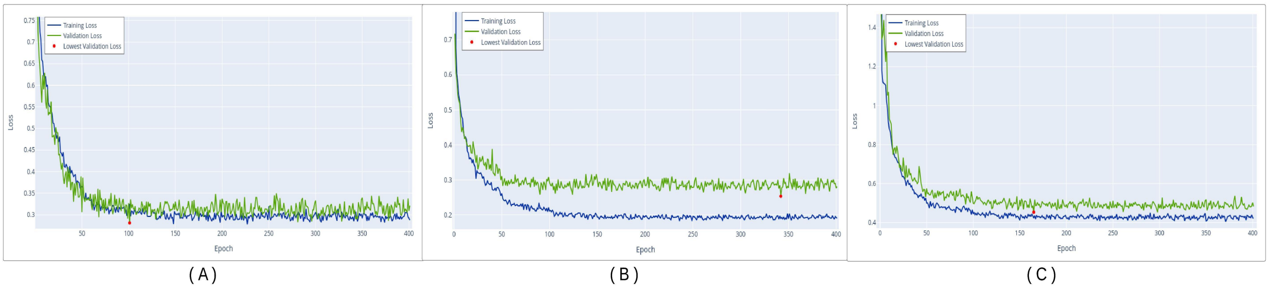

FIGURE 11. Number of epoch vs. training and validation loss (A. RSFE with EDED, B. DSFE with EDED and C. ESFE with EDED).

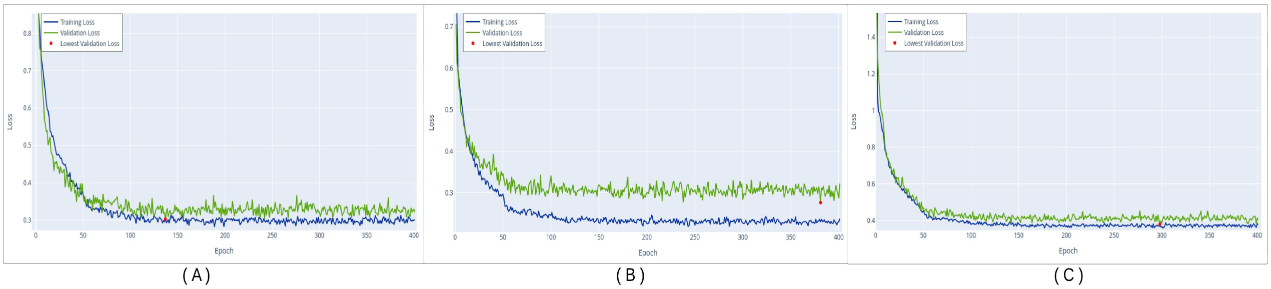

FIGURE 12. Number of epoch vs. training and validation loss (A. RSFE-PSN with EDED, B. DSFE-PSN with EDED and C. ESFE-PSN with EDED).

PSN into the base models, combined with EDED, consistently outperforms the other models, showcasing the effectiveness of our proposed methods in boosting classification accuracy, recall, and precision for Assamese script classification.

A performance growth analysis, as outlined in Table 2, highlights the comparative accuracy improvements from base models. In the five-way classification, the one-shot scenario notices accuracy gains of 15.49%, 22.26%, and 28.32% with EDED for RSFE, DSFE, and ESFE, respectively. Utilizing PSN and EDED together further enhances these gains to 24.57%, 42.15%, and 45.62%. Similar accuracy improvements were noticed across 5-shot, 10-shot, and 15-shot settings, revealing impactful and consistent performance enhancements. In the eight-way classification, the one-shot scenario shows accuracy growths of 31.78%, 21.27%, and 22.32% with EDED. Utilizing PSN and EDED together further boosts these gains to 40.77%, 33.28%, and 36.64%. The positive impact extends across 5-shot, 10-shot, and 15-shot scenarios.

Expanding our analysis to the 12-way classification, the 1-shot scenario gains accuracy growths of 21.79%, 22.03%, and 19.80% with EDED for RSFE, DSFE, and ESFE, respectively. Integrating PSN and EDED together further intensifies these gains to 35.14%, 37.63%, and 47.56%. Consistent improvements are noticeable across 5-shot, 10-shot, and 15-shot scenarios, highlighting the versatility and robustness of our proposed strategies in improving classification performance.

B. PERFORMANCE EVALUATION, METRICS AND ANALYSIS FOR TEST DATASET-2: OLCHIKI SCRIPT

This section evaluates the base models and their enhanced variants (utilizing EDED preprocessing and the PSN framework) on the Olchiki script utilizing different ways (5, 8, and 12) and shot (1, 5, and 10) configurations. Table 3 presents the performance metrics; each data point also derives from 10

TABLE 1. Comparison of Classification Accuracy, Recall and Precision for Assamese Scripts Across Various Ways and Shots, with Respect to RSFE, DSFE and ESFE.

			RSFE	RSFE +EDED	RSFE-PSN+EDED	DSFE	DSFE +EDED	DSFE-PSN+EDED	ESFE	ESFE +EDED	ESFE-PSN+EDED
5-way	1-shot	Accuracy	65.00 ± 9.00	75.07 ± 6.36	**80.97 ± 5.75**	63.51 ± 12.31	81.70 ± 7.65	**90.28 ± 4.70**	57.67 ± 8.99	74.03 ± 5.14	**83.98 ± 5.82**
		Recall	0.65 ± 0.09	0.75 ± 0.06	**0.80 ± 0.05**	0.64 ± 0.12	0.82 ± 0.08	**0.90 ± 0.05**	0.58 ± 0.09	0.73 ± 0.07	**0.84 ± 0.06**
		Precision	0.66 ± 0.08	0.76 ± 0.06	**0.82 ± 0.05**	0.66 ± 0.12	0.84 ± 0.07	**0.91 ± 0.05**	0.60 ± 0.09	0.74 ± 0.11	**0.86 ± 0.05**
	5-shot	Accuracy	74.86 ± 3.28	79.64 ± 4.41	**86.86 ± 3.53**	81.64 ± 6.68	89.00 ± 5.54	**92.69 ± 4.05**	74.75 ± 6.69	79.50 ± 6.96	**88.12 ± 4.22**
		Recall	0.74 ± 0.03	0.79 ± 0.04	**0.86 ± 0.03**	0.82 ± 0.07	0.89 ± 0.06	**0.93 ± 0.04**	0.75 ± 0.07	0.79 ± 0.07	**0.88 ± 0.04**
		Precision	0.77 ± 0.04	0.82 ± 0.04	**0.88 ± 0.02**	0.85 ± 0.04	0.91 ± 0.04	**0.93 ± 0.04**	0.79 ± 0.04	0.82 ± 0.06	**0.89 ± 0.04**
	10-shot	Accuracy	79.02 ± 6.71	83.27 ± 4.36	**88.98 ± 3.49**	85.55 ± 3.64	91.29 ± 4.78	**94.50 ± 3.88**	78.73 ± 5.19	80.86 ± 6.24	**89.86 ± 4.26**
		Recall	0.79 ± 0.06	0.83 ± 0.04	**0.88 ± 0.03**	0.86 ± 0.04	0.91 ± 0.05	**0.95 ± 0.04**	0.79 ± 0.05	0.81 ± 0.06	**0.90 ± 0.04**
		Precision	0.82 ± 0.05	0.84 ± 0.04	**0.90 ± 0.02**	0.87 ± 0.04	0.92 ± 0.04	**0.95 ± 0.03**	0.80 ± 0.05	0.83 ± 0.05	**0.91 ± 0.04**
	15-shot	Accuracy	80.76 ± 6.03	84.76 ± 3.84	**91.52 ± 2.69**	87.05 ± 3.69	93.25 ± 3.71	**95.83 ± 3.75**	78.75 ± 6.46	82.75 ± 5.93	**91.33 ± 3.95**
		Recall	0.80 ± 0.06	0.84 ± 0.03	**0.91 ± 0.02**	0.87 ± 0.04	0.93 ± 0.04	**0.96 ± 0.04**	0.79 ± 0.06	0.83 ± 0.06	**0.91 ± 0.04**
		Precision	0.81 ± 0.06	0.86 ± 0.02	**0.92 ± 0.02**	0.89 ± 0.03	0.94 ± 0.03	**0.96 ± 0.03**	0.80 ± 0.06	0.84 ± 0.03	**0.92 ± 0.04**
8-way	1-shot	Accuracy	50.60 ± 4.43	66.68 ± 8.34	**71.23 ± 7.55**	58.81 ± 11.21	74.70 ± 7.32	**78.38 ± 6.29**	49.64 ± 8.70	60.72 ± 3.03	**67.83 ± 4.46**
		Recall	0.50 ± 0.04	0.66 ± 0.08	**0.71 ± 0.07**	0.59 ± 0.11	0.75 ± 0.07	**0.78 ± 0.06**	0.50 ± 0.09	0.61 ± 0.03	**0.68 ± 0.04**
		Precision	0.50 ± 0.05	0.69 ± 0.08	**0.73 ± 0.07**	0.60 ± 0.12	0.77 ± 0.07	**0.81 ± 0.05**	0.52 ± 0.09	0.63 ± 0.04	**0.70 ± 0.04**
	5-shot	Accuracy	67.48 ± 6.95	74.85 ± 4.93	**78.27 ± 6.07**	74.07 ± 5.19	82.93 ± 6.28	**86.38 ± 4.73**	61.00 ± 7.31	65.88 ± 3.79	**74.60 ± 5.51**
		Recall	0.67 ± 0.06	0.74 ± 0.04	**0.78 ± 0.06**	0.74 ± 0.05	0.83 ± 0.06	**0.86 ± 0.05**	0.61 ± 0.07	0.66 ± 0.04	**0.75 ± 0.06**
		Precision	0.69 ± 0.07	0.76 ± 0.05	**0.80 ± 0.05**	0.77 ± 0.04	0.84 ± 0.07	**0.87 ± 0.05**	0.62 ± 0.09	0.67 ± 0.04	**0.76 ± 0.06**
	10-shot	Accuracy	70.71 ± 5.28	76.75 ± 5.12	**80.42 ± 5.32**	77.88 ± 4.71	84.16 ± 7.19	**87.55 ± 5.00**	64.06 ± 6.54	68.80 ± 3.93	**76.63 ± 5.50**
		Recall	0.70 ± 0.05	0.76 ± 0.05	**0.80 ± 0.05**	0.78 ± 0.05	0.84 ± 0.07	**0.88 ± 0.05**	0.64 ± 0.07	0.69 ± 0.04	**0.77 ± 0.06**
		Precision	0.74 ± 0.03	0.77 ± 0.04	**0.81 ± 0.04**	0.80 ± 0.04	0.85 ± 0.06	**0.88 ± 0.05**	0.65 ± 0.08	0.70 ± 0.06	**0.78 ± 0.06**
	15-shot	Accuracy	73.35 ± 4.36	78.61 ± 4.86	**81.60 ± 4.89**	79.68 ± 3.95	85.90 ± 6.14	**88.98 ± 4.21**	64.70 ± 7.17	70.20 ± 4.23	**78.37 ± 5.04**
		Recall	0.73 ± 0.04	0.78 ± 0.04	**0.81 ± 0.04**	0.80 ± 0.04	0.86 ± 0.06	**0.89 ± 0.04**	0.65 ± 0.07	0.70 ± 0.04	**0.78 ± 0.05**
		Precision	0.75 ± 0.06	0.80 ± 0.04	**0.83 ± 0.04**	0.82 ± 0.04	0.87 ± 0.06	**0.89 ± 0.04**	0.66 ± 0.10	0.72 ± 0.05	**0.80 ± 0.05**
12-way	1-shot	Accuracy	49.83 ± 5.55	60.69 ± 3.36	**67.34 ± 2.67**	53.17 ± 7.82	68.19 ± 4.49	**73.18 ± 4.25**	41.32 ± 3.07	49.50 ± 4.18	**60.97 ± 4.07**
		Recall	0.49 ± 0.05	0.60 ± 0.03	**0.67 ± 0.02**	0.53 ± 0.08	0.68 ± 0.04	**0.73 ± 0.04**	0.41 ± 0.03	0.49 ± 0.04	**0.61 ± 0.04**
		Precision	0.51 ± 0.04	0.61 ± 0.04	**0.70 ± 0.03**	0.56 ± 0.08	0.70 ± 0.05	**0.75 ± 0.04**	0.43 ± 0.06	0.51 ± 0.04	**0.64 ± 0.05**
	5-shot	Accuracy	63.40 ± 3.79	70.43 ± 3.11	**74.74 ± 4.08**	69.47 ± 5.72	81.02 ± 3.54	**83.65 ± 2.32**	53.80 ± 2.94	59.34 ± 2.74	**71.86 ± 3.84**
		Recall	0.63 ± 0.03	0.70 ± 0.03	**0.74 ± 0.04**	0.69 ± 0.06	0.81 ± 0.04	**0.84 ± 0.02**	0.54 ± 0.03	0.59 ± 0.03	**0.72 ± 0.04**
		Precision	0.67 ± 0.05	0.72 ± 0.04	**0.76 ± 0.04**	0.74 ± 0.06	0.82 ± 0.04	**0.85 ± 0.03**	0.56 ± 0.04	0.61 ± 0.04	**0.74 ± 0.04**
	10-shot	Accuracy	67.57 ± 5.69	72.71 ± 2.80	**76.87 ± 3.87**	73.96 ± 4.72	82.86 ± 3.53	**85.30 ± 2.41**	57.73 ± 3.17	61.38 ± 2.96	**74.25 ± 3.59**
		Recall	0.67 ± 0.05	0.72 ± 0.02	**0.76 ± 0.03**	0.74 ± 0.05	0.83 ± 0.04	**0.85 ± 0.02**	0.58 ± 0.03	0.61 ± 0.03	**0.74 ± 0.04**
		Precision	0.70 ± 0.06	0.74 ± 0.03	**0.78 ± 0.04**	0.77 ± 0.04	0.84 ± 0.04	**0.86 ± 0.03**	0.60 ± 0.03	0.62 ± 0.04	**0.76 ± 0.04**
	15-shot	Accuracy	69.81 ± 3.77	74.54 ± 2.63	**78.61 ± 3.96**	76.84 ± 3.90	84.12 ± 2.46	**86.84 ± 2.50**	59.97 ± 2.68	63.89 ± 3.45	**75.55 ± 3.82**
		Recall	0.69 ± 0.03	0.74 ± 0.02	**0.78 ± 0.03**	0.77 ± 0.04	0.84 ± 0.02	**0.87 ± 0.02**	0.60 ± 0.03	0.64 ± 0.03	**0.76 ± 0.04**
		Precision	0.73 ± 0.02	0.76 ± 0.02	**0.80 ± 0.03**	0.79 ± 0.04	0.85 ± 0.02	**0.88 ± 0.02**	0.61 ± 0.04	0.65 ± 0.05	**0.77 ± 0.04**

TABLE 2. Performance Gain Comparison with EDED and Integration of PSN for RSFE, DSFE, and ESFE on Assamese Script.

		RSFE		DSFE		ESFE	
		EDED	**PSN+ EDED**	**EDED**	**PSN+ EDED**	**EDED**	**PSN+ EDED**
5 - way Accuracy Gain (%)	1-shot	15.49	24.57	22.26	42.15	28.32	45.62
	5-shot	6.39	16.03	8.27	13.54	6.35	17.89
	10-shot	5.38	12.60	6.29	10.46	2.71	14.14
	15-shot	4.95	13.32	6.65	10.09	5.08	15.97
8 - way Accuracy Gain (%)	1-shot	31.78	40.77	21.27	33.28	22.32	36.64
	5-shot	10.92	15.99	10.68	16.62	8.00	22.30
	10-shot	8.54	13.73	7.46	12.42	7.40	19.62
	15-shot	7.17	11.25	7.24	11.67	8.50	21.13
12 - way Accuracy Gain (%)	1-shot	21.79	35.14	22.03	37.63	19.80	47.56
	5-shot	11.09	17.89	14.26	20.41	10.30	33.57
	10-shot	7.61	13.76	10.74	15.33	6.32	28.62
	15-shot	6.78	12.61	8.65	13.01	6.54	25.98

independent testing assignments. Our approach consistently evaluates accuracy, recall, and precision to demonstrate the effectiveness of EDED and PSN in Olchiki script classification.

An in-depth analysis of the performance improvement, as outlined in Table 4, highlights accuracy growths in the one-shot scenario for five-way classification with EDED, the base models are acquired gains of 51.75%, 49.28%, and 5.86% for RSFE, DSFE, and ESFE, respectively. Utilizing PSN and EDED together further boosts these gains to 64.53%, 60.30%, and 40.20%. A similar accuracy improvments persists across other shot scenarios. In the eight-way classification, EDED returns one-shot gains of 58.62%, 66.69%, and 4.21%, with PSN and EDED together further enhancing to 91.58%, 77.08%, and 60.84%. However, some examples of accuracy loss highlight the need for fair optimization in our strategies. Expanding to the 12-way classification, one-shot proceeds of 66.41%, 85.41%, and 5.99% accuracy gain with EDED, which is further extended to 106.78%, 105.84%, and 88.84% with PSN and EDED together. Despite a few accuracy losses, the overall impact highlights the strength of our approach.

C. PERFORMANCE EVALUATION, METRICS AND ANALYSIS FOR TEST DATASET-3: MAPI SCRIPT:

This section evaluates base models and their refined versions, combining EDED preprocessing and our proposed PSN model for Mapi script classification. Table 5 depicts significant performance growths across 5-way, 8-way, and 12-way classifications throughout diverse shot scenarios.

TABLE 3. Comparison of Classification Recall and Precision for Olchiki Scripts Across Various Ways and Shots, with Respect to RSFE, DSFE and ESFE.

			RSFE			DSFE			ESFE		
			RSFE	RSFE +EDED	RSFE-PSN+EDED	DSFE	DSFE +EDED	DSFE-PSN+EDED	ESFE	ESFE +EDED	ESFE-PSN+EDED
5-way	1-shot	Accuracy	48.52 ±7.69	73.63 ±8.51	**79.83 ±10.03**	54.38 ±6.30	81.18 ±9.34	**87.17 ±9.08**	56.44 ±6.28	59.75 ±6.95	**79.13 ±9.07**
		Precision	0.51 ±0.09	0.76 ±0.10	**0.82 ±0.10**	0.57 ±0.06	0.82 ±0.09	**0.88 ±0.09**	0.59 ±0.09	0.62 ±0.06	**0.80 ±0.09**
		Recall	0.49 ±0.08	0.74 ±0.09	**0.80 ±0.10**	0.54 ±0.06	0.81 ±0.09	**0.87 ±0.09**	0.56 ±0.06	0.60 ±0.07	**0.79 ±0.09**
	5-shot	Accuracy	52.05 ±8.97	76.43 ±8.26	**83.02 ±9.63**	60.16 ±5.79	84.23 ±8.35	**88.73 ±8.35**	60.65 ±6.05	64.52 ±5.43	**84.40 ±8.20**
		Precision	0.52 ±0.11	0.79 ±0.09	**0.84 ±0.10**	0.64 ±0.05	0.84 ±0.10	**0.89 ±0.09**	0.63 ±0.08	0.67 ±0.07	**0.85 ±0.09**
		Recall	0.52 ±0.09	0.76 ±0.08	**0.83 ±0.10**	0.60 ±0.06	0.84 ±0.08	**0.89 ±0.08**	0.61 ±0.06	0.65 ±0.05	**0.84 ±0.08**
	10-shot	Accuracy	54.57 ±8.41	78.10 ±7.71	**83.83 ±9.45**	63.68 ±5.69	85.30 ±8.23	**89.60 ±8.60**	62.62 ±5.24	65.88 ±5.49	**85.59 ±8.41**
		Precision	0.54 ±0.10	0.80 ±0.08	**0.85 ±0.10**	0.66 ±0.07	0.86 ±0.08	**0.90 ±0.09**	0.65 ±0.06	0.71 ±0.07	**0.86 ±0.08**
		Recall	0.55 ±0.08	0.78 ±0.08	**0.84 ±0.09**	0.64 ±0.06	0.85 ±0.08	**0.90 ±0.09**	0.63 ±0.05	0.66 ±0.05	**0.86 ±0.08**
	15-shot	Accuracy	55.45 ±8.26	79.32 ±8.25	**85.29 ±9.38**	65.53 ±5.08	87.00 ±8.25	**90.53 ±8.99**	63.42 ±5.58	66.90 ±4.84	**86.57 ±8.42**
		Precision	0.57 ±0.12	0.79 ±0.11	**0.85 ±0.12**	0.68 ±0.06	0.86 ±0.11	**0.91 ±0.09**	0.65 ±0.07	0.70 ±0.06	**0.87 ±0.09**
		Recall	0.55 ±0.08	0.79 ±0.08	**0.85 ±0.09**	0.66 ±0.05	0.87 ±0.08	**0.91 ±0.09**	0.63 ±0.06	0.67 ±0.05	**0.87 ±0.08**
8-way	1-shot	Accuracy	37.65 ±5.32	59.72 ±5.12	**72.13 ±4.85**	45.90 ±5.56	76.51 ±7.40	**81.28 ±7.08**	45.58 ±7.85	47.50 ±3.09	**73.31 ±6.18**
		Precision	0.39 ±0.07	0.63 ±0.05	**0.73 ±0.07**	0.49 ±0.06	0.79 ±0.07	**0.84 ±0.07**	0.48 ±0.09	0.49 ±0.03	**0.75 ±0.06**
		Recall	0.38 ±0.05	0.60 ±0.05	**0.72 ±0.05**	0.46 ±0.06	0.77 ±0.07	**0.81 ±0.07**	0.46 ±0.08	0.47 ±0.03	**0.73 ±0.06**
	5-shot	Accuracy	41.13 ±4.96	62.11 ±5.84	**75.88 ±5.79**	51.43 ±5.10	81.66 ±6.59	**86.55 ±7.66**	51.23 ±7.93	52.63 ±2.33	**79.55 ±5.84**
		Precision	0.41 ±0.06	0.66 ±0.05	**0.79 ±0.05**	0.54 ±0.08	0.84 ±0.06	**0.87 ±0.08**	0.56 ±0.07	0.52 ±0.07	**0.82 ±0.06**
		Recall	0.41 ±0.05	0.62 ±0.06	**0.76 ±0.06**	0.51 ±0.05	0.82 ±0.07	**0.87 ±0.08**	0.51 ±0.08	0.53 ±0.02	**0.80 ±0.06**
	10-shot	Accuracy	42.83 ±4.90	63.42 ±5.63	**78.35 ±5.81**	54.48 ±4.96	83.58 ±6.32	**88.10 ±6.55**	53.93 ±7.34	55.51 ±2.66	**82.24 ±5.64**
		Precision	0.40 ±0.06	0.68 ±0.04	**0.80 ±0.06**	0.59 ±0.05	0.84 ±0.08	**0.89 ±0.06**	0.57 ±0.08	0.55 ±0.06	**0.83 ±0.06**
		Recall	0.43 ±0.05	0.63 ±0.06	**0.78 ±0.06**	0.54 ±0.05	0.84 ±0.06	**0.88 ±0.07**	0.54 ±0.07	0.56 ±0.03	**0.82 ±0.06**
	15-shot	Accuracy	44.19 ±4.28	64.56 ±5.94	**79.87 ±5.61**	56.36 ±4.57	84.24 ±6.53	**88.67 ±6.58**	55.66 ±7.36	55.42 ±2.31	**83.48 ±5.80**
		Precision	0.42 ±0.06	0.69 ±0.06	**0.82 ±0.06**	0.59 ±0.04	0.85 ±0.08	**0.89 ±0.06**	0.60 ±0.06	0.54 ±0.05	**0.84 ±0.06**
		Recall	0.44 ±0.04	0.65 ±0.06	**0.80 ±0.06**	0.56 ±0.05	0.84 ±0.07	**0.89 ±0.07**	0.56 ±0.07	0.55 ±0.02	**0.83 ±0.06**
12-way	1-shot	Accuracy	28.19 ±2.31	46.91 ±6.12	**58.29 ±6.77**	34.76 ±2.99	64.45 ±4.18	**71.55 ±4.99**	32.07 ±3.52	33.99 ±3.80	**60.56 ±5.23**
		Precision	0.31 ±0.04	0.49 ±0.07	**0.60 ±0.06**	0.38 ±0.04	0.67 ±0.04	**0.74 ±0.04**	0.35 ±0.04	0.36 ±0.03	**0.63 ±0.05**
		Recall	0.28 ±0.02	0.47 ±0.06	**0.58 ±0.07**	0.35 ±0.03	0.64 ±0.04	**0.72 ±0.05**	0.32 ±0.04	0.34 ±0.04	**0.61 ±0.05**
	5-shot	Accuracy	31.52 ±1.48	50.53 ±5.88	**64.03 ±5.66**	38.23 ±2.08	70.89 ±3.25	**76.95 ±4.98**	37.85 ±1.84	37.68 ±3.12	**69.26 ±4.52**
		Precision	0.30 ±0.05	0.54 ±0.05	**0.67 ±0.05**	0.42 ±0.03	0.75 ±0.03	**0.79 ±0.05**	0.40 ±0.03	0.35 ±0.07	**0.71 ±0.06**
		Recall	0.32 ±0.01	0.51 ±0.06	**0.64 ±0.06**	0.38 ±0.02	0.71 ±0.03	**0.77 ±0.05**	0.38 ±0.02	0.38 ±0.03	**0.69 ±0.05**
	10-shot	Accuracy	33.19 ±2.27	51.94 ±6.43	**66.38 ±5.93**	40.55 ±1.89	72.71 ±3.17	**77.86 ±5.29**	39.91 ±2.22	40.52 ±1.81	**71.10 ±4.57**
		Precision	0.33 ±0.05	0.55 ±0.07	**0.69 ±0.06**	0.44 ±0.04	0.74 ±0.06	**0.79 ±0.05**	0.42 ±0.04	0.35 ±0.04	**0.73 ±0.05**
		Recall	0.33 ±0.02	0.52 ±0.06	**0.66 ±0.06**	0.41 ±0.02	0.73 ±0.03	**0.78 ±0.05**	0.40 ±0.02	0.41 ±0.02	**0.71 ±0.05**
	15-shot	Accuracy	34.84 ±2.30	54.05 ±5.92	**67.72 ±5.69**	42.29 ±1.91	74.07 ±3.16	**79.21 ±4.75**	41.36 ±2.05	41.78 ±1.28	**72.56 ±4.87**
		Precision	0.32 ±0.04	0.55 ±0.07	**0.70 ±0.05**	0.45 ±0.05	0.75 ±0.05	**0.80 ±0.05**	0.43 ±0.05	0.37 ±0.04	**0.73 ±0.06**
		Recall	0.35 ±0.02	0.54 ±0.06	**0.68 ±0.06**	0.42 ±0.02	0.74 ±0.03	**0.79 ±0.05**	0.41 ±0.02	0.42 ±0.01	**0.73 ±0.05**

TABLE 4. Performance Gain Comparison with EDED and Integration of PSN for RSFE, DSFE, and ESFE on olchiki Script.

		RSFE		DSFE		ESFE	
		EDED	**PSN+ EDED**	**EDED**	**PSN+ EDED**	**EDED**	**PSN+ EDED**
5 - way Accuracy Gain (%)	1-shot	51.75	64.53	49.28	60.30	5.86	40.20
	5-shot	46.84	59.50	40.01	47.49	6.38	39.16
	10-shot	43.12	53.62	33.95	40.70	5.21	36.68
	15-shot	43.05	53.81	32.76	38.15	5.49	36.50
8 - way Accuracy Gain (%)	1-shot	58.62	91.58	66.69	77.08	4.21	60.84
	5-shot	51.01	84.49	58.78	68.29	2.73	55.28
	10-shot	48.07	82.93	53.41	61.71	2.93	52.49
	15-shot	46.10	80.74	49.47	57.33	-0.43	49.98
12 - way Accuracy Gain (%)	1-shot	66.41	106.78	85.41	105.84	5.99	88.84
	5-shot	60.31	103.14	85.43	101.28	-0.45	82.99
	10-shot	56.49	100.00	79.31	92.01	1.53	78.15
	15-shot	55.14	94.37	75.15	87.30	1.02	75.44

An analysis of the performance improvement, as outlined in Table 6, highlights accuracy growth throughout all configurations. In the five-way classification, the base models provides accuracy boosts of 12%, 35.22%, and 9.27% with EDED, escalating those gains to 20.10%, 47.47%, and 30.85% when PSN is integrated with EDED. This favorable trend of accuracy boosts continues consistently throughout 5-shot, 10-shot, and 15-shot scenarios. For the 8-way classification, the impact of EDED is evident, with 13.86%, 39.82%, and 7.87% accuracy gains in the one-shot setting. Eventually, the integration of PSN boosts these gains to 22.04%, 47.48%, and 30.99%. Extending our examination to the challenging 12-way classification task, our models achieve one-shot gains of 14.28%, 66.94%, and 5.31% with EDED, which further improves to 26.19%, 79.71%, and 58.00% with PSN and EDED together. The similar performance improvements are observed across 5-shot, 10-shot, and 15-shot scenarios. Occasional accuracy losses are recorded, especially in 5-shot and 10-shot scenarios for the 5-way and 8-way classifications. In these cases, a slight reduction in accuracy is observed, which provides the scopes for further optimization. Regardless, our strategy consistently delivers significant overall performance improvement, highlights the strength of our approach.

VI. CONCLUSION

In this paper, we proposed a novel methodology in low-resource regional language OCR tasks using a Siamese meta-learning strategy. The methodology includes a novel priority smart network (PSN) framework and a novel preprocessing, called enhanced differential edge detection (EDED) to

TABLE 5. Comparison of Classification Accuracy Recall and Precision for mapi Scripts Across Various Ways and Shots, with Respect to RSFE, DSFE and ESFE.

			RSFE			DSFE			ESFE		
5-way	1-shot	Accuracy	60.14 ± 4.97	67.36 ± 7.11	**72.23 ± 6.91**	57.46 ± 2.98	77.7 ± 6.13	**84.74 ± 7.36**	55.62 ± 4.58	60.78 ± 6.4	**72.78 ± 6.08**
		Precision	0.6 ± 0.06	0.71 ± 0.07	**0.75 ± 0.06**	0.61 ± 0.03	0.8 ± 0.05	**0.87 ± 0.06**	0.59 ± 0.05	0.64 ± 0.06	**0.75 ± 0.06**
		Recall	0.6 ± 0.05	0.67 ± 0.07	**0.72 ± 0.07**	0.57 ± 0.03	0.78 ± 0.06	**0.85 ± 0.07**	0.56 ± 0.05	0.61 ± 0.06	**0.73 ± 0.06**
	5-shot	Accuracy	70.63 ± 5.37	75.54 ± 4.62	**82.07 ± 6.02**	66.36 ± 8.41	82.81 ± 5.74	**88.58 ± 5.99**	67.15 ± 4.74	63.85 ± 5.2	**80.8 ± 3.35**
		Precision	0.73 ± 0.06	0.78 ± 0.04	**0.83 ± 0.06**	0.75 ± 0.08	0.85 ± 0.06	**0.89 ± 0.06**	0.7 ± 0.05	0.65 ± 0.08	**0.82 ± 0.03**
		Recall	0.71 ± 0.05	0.76 ± 0.05	**0.82 ± 0.06**	0.66 ± 0.08	0.83 ± 0.06	**0.89 ± 0.06**	0.67 ± 0.05	0.64 ± 0.05	**0.81 ± 0.03**
	10-shot	Accuracy	74.75 ± 5.03	77.81 ± 4.99	**84.28 ± 5.48**	70.65 ± 8.48	85.5 ± 5.58	**90.13 ± 5.03**	69.26 ± 6.56	64.84 ± 4.94	**82.32 ± 3.15**
		Precision	0.78 ± 0.05	0.79 ± 0.05	**0.85 ± 0.05**	0.74 ± 0.07	0.86 ± 0.05	**0.91 ± 0.05**	0.73 ± 0.06	0.64 ± 0.06	**0.84 ± 0.02**
		Recall	0.75 ± 0.05	0.78 ± 0.05	**0.84 ± 0.05**	0.71 ± 0.08	0.86 ± 0.06	**0.9 ± 0.05**	0.69 ± 0.07	0.65 ± 0.05	**0.82 ± 0.03**
	15-shot	Accuracy	77.11 ± 4.62	79.76 ± 3.79	**86.27 ± 5.22**	72.64 ± 8.61	87.32 ± 5.38	**91.23 ± 4.76**	71.49 ± 6.08	66.9 ± 4.17	**84.25 ± 3.36**
		Precision	0.79 ± 0.05	0.81 ± 0.04	**0.87 ± 0.05**	0.77 ± 0.07	0.88 ± 0.05	**0.92 ± 0.04**	0.74 ± 0.05	0.67 ± 0.07	**0.85 ± 0.03**
		Recall	0.77 ± 0.05	0.8 ± 0.04	**0.86 ± 0.05**	0.73 ± 0.09	0.87 ± 0.05	**0.91 ± 0.05**	0.71 ± 0.06	0.67 ± 0.04	**0.84 ± 0.03**
8-way	1-shot	Accuracy	51.95 ± 6.39	59.15 ± 4.62	**63.40 ± 6.22**	49.12 ± 7.95	68.68 ± 3.40	**72.44 ± 3.85**	46.28 ± 6.74	49.92 ± 8.33	**60.62 ± 3.95**
		Precision	0.55 ± 0.06	0.62 ± 0.06	**0.67 ± 0.08**	0.56 ± 0.09	0.71 ± 0.03	**0.74 ± 0.04**	0.46 ± 0.08	0.53 ± 0.08	**0.62 ± 0.04**
		Recall	0.52 ± 0.06	0.59 ± 0.05	**0.63 ± 0.06**	0.49 ± 0.08	0.69 ± 0.03	**0.72 ± 0.04**	0.46 ± 0.07	0.50 ± 0.08	**0.61 ± 0.04**
	5-shot	Accuracy	64.83 ± 2.86	66.46 ± 3.17	**71.60 ± 5.17**	63.70 ± 2.45	79.95 ± 2.30	**82.36 ± 2.93**	58.67 ± 4.21	60.66 ± 4.11	**70.55 ± 2.62**
		Precision	0.67 ± 0.04	0.67 ± 0.04	**0.74 ± 0.05**	0.68 ± 0.05	0.81 ± 0.02	**0.84 ± 0.02**	0.63 ± 0.04	0.63 ± 0.03	**0.74 ± 0.02**
		Recall	0.65 ± 0.03	0.66 ± 0.03	**0.72 ± 0.05**	0.64 ± 0.02	0.80 ± 0.02	**0.82 ± 0.03**	0.59 ± 0.04	0.61 ± 0.04	**0.71 ± 0.03**
	10-shot	Accuracy	66.81 ± 3.09	68.87 ± 3.44	**74.28 ± 5.10**	66.92 ± 3.91	82.04 ± 2.18	**84.77 ± 3.22**	61.65 ± 4.26	62.70 ± 4.44	**72.26 ± 2.73**
		Precision	0.71 ± 0.05	0.70 ± 0.05	**0.75 ± 0.05**	0.73 ± 0.03	0.83 ± 0.02	**0.86 ± 0.03**	0.64 ± 0.04	0.65 ± 0.04	**0.74 ± 0.02**
		Recall	0.67 ± 0.03	0.69 ± 0.03	**0.74 ± 0.05**	0.67 ± 0.04	0.82 ± 0.02	**0.85 ± 0.03**	0.62 ± 0.04	0.63 ± 0.04	**0.72 ± 0.03**
	15-shot	Accuracy	69.25 ± 3.15	70.66 ± 2.59	**74.91 ± 5.05**	69.06 ± 3.59	83.99 ± 1.38	**86.57 ± 3.06**	62.79 ± 4.79	63.76 ± 4.72	**73.90 ± 3.08**
		Precision	0.73 ± 0.04	0.71 ± 0.04	**0.76 ± 0.05**	0.73 ± 0.04	0.85 ± 0.02	**0.87 ± 0.03**	0.65 ± 0.04	0.66 ± 0.04	**0.75 ± 0.03**
		Recall	0.69 ± 0.03	0.71 ± 0.03	**0.75 ± 0.05**	0.69 ± 0.04	0.84 ± 0.01	**0.87 ± 0.03**	0.63 ± 0.05	0.64 ± 0.05	**0.74 ± 0.03**
12-way	1-shot	Accuracy	43.42 ± 5.57	49.62 ± 4.00	**54.79 ± 4.12**	35.97 ± 3.33	60.05 ± 5.19	**64.64 ± 1.78**	33.52 ± 3.33	35.30 ± 5.19	**52.96 ± 1.78**
		Precision	0.46 ± 0.06	0.53 ± 0.03	**0.58 ± 0.46**	0.40 ± 0.03	0.63 ± 0.04	**0.67 ± 0.03**	0.36 ± 0.03	0.36 ± 0.04	**0.54 ± 0.03**
		Recall	0.43 ± 0.05	0.50 ± 0.04	**0.55 ± 0.41**	0.36 ± 0.03	0.60 ± 0.05	**0.65 ± 0.01**	0.34 ± 0.03	0.35 ± 0.05	**0.53 ± 0.01**
	5-shot	Accuracy	53.74 ± 4.00	58.36 ± 2.04	**64.26 ± 3.56**	47.61 ± 4.27	72.32 ± 4.21	**77.45 ± 2.16**	43.82 ± 1.61	46.71 ± 4.83	**62.61 ± 2.71**
		Precision	0.60 ± 0.05	0.60 ± 0.03	**0.67 ± 0.03**	0.54 ± 0.05	0.75 ± 0.04	**0.79 ± 0.02**	0.44 ± 0.03	0.48 ± 0.07	**0.66 ± 0.02**
		Recall	0.54 ± 0.04	0.58 ± 0.02	**0.64 ± 0.03**	0.48 ± 0.04	0.72 ± 0.04	**0.77 ± 0.02**	0.44 ± 0.01	0.47 ± 0.04	**0.63 ± 0.02**
	10-shot	Accuracy	56.85 ± 4.41	60.54 ± 2.20	**67.46 ± 2.50**	52.75 ± 3.15	75.26 ± 3.87	**80.48 ± 2.77**	48.25 ± 2.51	49.36 ± 5.37	**64.55 ± 2.98**
		Precision	0.61 ± 0.06	0.64 ± 0.03	**0.70 ± 0.03**	0.60 ± 0.05	0.78 ± 0.03	**0.82 ± 0.02**	0.49 ± 0.04	0.49 ± 0.07	**0.67 ± 0.03**
		Recall	0.57 ± 0.04	0.61 ± 0.02	**0.67 ± 0.02**	0.53 ± 0.03	0.75 ± 0.03	**0.80 ± 0.02**	0.48 ± 0.02	0.49 ± 0.05	**0.65 ± 0.02**
	15-shot	Accuracy	58.56 ± 3.56	61.80 ± 2.07	**69.14 ± 3.15**	57.20 ± 4.70	76.72 ± 3.55	**81.43 ± 2.38**	49.41 ± 2.78	51.10 ± 3.56	**65.99 ± 3.04**
		Precision	0.63 ± 0.04	0.64 ± 0.03	**0.71 ± 0.03**	0.65 ± 0.04	0.78 ± 0.03	**0.82 ± 0.02**	0.51 ± 0.04	0.52 ± 0.06	**0.68 ± 0.03**
		Recall	0.59 ± 0.03	0.62 ± 0.02	**0.69 ± 0.03**	0.57 ± 0.04	0.77 ± 0.03	**0.81 ± 0.02**	0.49 ± 0.02	0.51 ± 0.03	**0.66 ± 0.03**

TABLE 6. Performance Gain Comparison with EDED and Integration of PSN for RSFE, DSFE, and ESFE on mapi dataset.

		RSFE		DSFE		ESFE	
		EDED	**PSN+ EDED**	**EDED**	**PSN+ EDED**	**EDED**	**PSN+ EDED**
5- way Accuracy Gain (%)	1-shot	12	20.10	35.22	47.47	9.27	30.85
	5-shot	6.95	16.19	24.78	33.48	-4.91	20.32
	10-shot	4.09	12.74	21.01	27.57	-6.38	18.85
	15-shot	3.43	11.87	20.20	25.59	-6.42	17.84
8 - way Accuracy Gain (%)	1-shot	13.86	22.04	39.82	47.48	7.87	30.99
	5-shot	2.51	10.44	25.51	29.29	3.39	20.25
	10-shot	3.08	11.18	22.59	26.67	1.70	17.21
	15-shot	2.04	8.17	21.62	25.35	1.54	17.69
12 - way Accuracy Gain (%)	1-shot	14.28	26.19	66.94	79.71	5.31	58.00
	5-shot	8.60	19.58	51.90	62.68	6.60	42.88
	10-shot	6.49	18.66	42.67	52.57	2.30	33.78
	15-shot	5.53	18.07	34.13	42.36	3.42	33.56

improve performance. The EDED strategy significantly enhanced the performance of OCR classification. We extensively experimented using three benchmark OCR datasets and three state-of-the-art models (ResNet, DensNet, and EfficientNet). Our results reveal that the EDED strategy significantly improves the performance of the base models. The accuracy is further boosted to a new level when combined with the PSN framework. However, it is essential to note that in some rare cases, the EDED technique's reduced performance corresponded to the base models. Further research can be conducted to address the rare cases where the EDED strategy reduces accuracy and to investigate its applications in other fields.

REFERENCES

[1] G. L. Cash and M. Hatamian, "Optical character recognition by the method of moments," *Computer Vision, Graphics, and Image Processing*, vol. 39, no. 3, pp. 291–310, 1987.

[2] J. Memon, M. Sami, R. A. Khan, and M. Uddin, "Handwritten optical character recognition (ocr): A comprehensive systematic literature review (slr)," *IEEE access*, vol. 8, pp. 142642–142668, 2020.

[3] Y. Wu, H. Bi, J. Fan, G. Xu, and H. Chen, "Dmhc: Device-free multimodal handwritten character recognition system with acoustic signal," *Knowledge-Based Systems*, vol. 264, p. 110314, 2023.

[4] B. Harish and R. K. Rangan, "A comprehensive survey on indian regional language processing," *SN Applied Sciences*, vol. 2, no. 7, p. 1204, 2020.

[5] R. Buoy, M. Iwamura, S. Srun, and K. Kise, "Toward a low-resource non-latin-complete baseline: An exploration of khmer optical character recognition," *IEEE Access*, vol. 11, pp. 128044–128060, 2023.

[6] S. Ranathunga, E.-S. A. Lee, M. Prifti Skenduli, R. Shekhar, M. Alam,

and R. Kaur, "Neural machine translation for low-resource languages: A survey," *ACM Computing Surveys*, vol. 55, no. 11, pp. 1–37, 2023.

[7] S. Rijhwani, *Improving Optical Character Recognition for Endangered Languages*. PhD thesis, Carnegie Mellon University, 2023.

[8] A. Sufian, A. Ghosh, A. Naskar, F. Sultana, J. Sil, and M. H. Rahman, "Bdnet: bengali handwritten numeral digit recognition based on densely connected convolutional neural networks," *Journal of King Saud University-Computer and Information Sciences*, vol. 34, no. 6, pp. 2610–2620, 2022.

[9] M. Pei, B. Yan, H. Hao, and M. Zhao, "Person-specific face spoofing detection based on a siamese network," *Pattern Recognition*, vol. 135, p. 109148, 2023.

[10] F. Yan and H. Zhang, "Smfnet: One-shot recognition of chinese character font based on siamese metric model," *IEEE Access*, vol. 12, pp. 38473–38489, 2024.

[11] M. E. Vasconcellos, B. G. Ferreira, J. S. Leandro, B. F. Neto, F. R. Cordeiro, I. A. Cestari, M. A. Gutierrez, A. Sobrinho, and T. D. Cordeiro, "Siamese convolutional neural network for heartbeat classification using limited 12-lead ecg datasets," *Ieee Access*, vol. 11, pp. 5365–5376, 2023.

[12] S. Singh, P. K. Sarangi, C. Singla, and A. K. Sahoo, "Odia character recognition system: A study on feature extraction and classification techniques," *Materials Today: Proceedings*, vol. 34, pp. 742–747, 2021. 3rd International Conference on Science and Engineering in Materials.

[13] R. Zitouni, H. Bezine, and N. Arous, "Template matching and deep cnn-svm for online characters recognition," in *Intelligent Systems Design and Applications* (A. Abraham, V. Piuri, N. Gandhi, P. Siarry, A. Kaklauskas, and A. Madureira, eds.), (Cham), pp. 891–900, Springer International Publishing, 2021.

[14] Y. B. Hamdan and A. Sathesh, "Construction of statistical svm based recognition model for handwritten character recognition," *Journal of Information Technology*, vol. 3, no. 02, pp. 92–107, 2021.

[15] A. K. Bhunia, S. Ghose, A. Kumar, P. N. Chowdhury, A. Sain, and Y.-Z. Song, "Metahtr: Towards writer-adaptive handwritten text recognition," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 15830–15839, 2021.

[16] A. Naseer and K. Zafar, "Meta-feature based few-shot siamese learning for urdu optical character recognition," *Computational Intelligence*, vol. 38, no. 5, pp. 1707–1727, 2022.

[17] M. H. Talbo, H. Wang, L. Chi, and Y.-P. P. Chen, "Validating siamese embedded neural networks with identical representations for efficient model convergence," *Knowledge-Based Systems*, vol. 286, p. 111379, 2024.

[18] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *International conference on machine learning*, pp. 1126–1135, PMLR, 2017.

[19] A. Santoro, S. Bartunov, M. Botvinick, D. Wierstra, and T. Lillicrap, "Meta-learning with memory-augmented neural networks," in *International conference on machine learning*, pp. 1842–1850, PMLR, 2016.

[20] W. Han, X. Ren, H. Lin, Y. Fu, and X. Xue, "Self-supervised learning of orc-bert augmentor for recognizing few-shot oracle characters," in *Computer Vision – ACCV 2020: 15th Asian Conference on Computer Vision, Kyoto, Japan, November 30 – December 4, 2020, Revised Selected Papers, Part VI*, (Berlin, Heidelberg), p. 652–668, Springer-Verlag, 2020.

[21] C.-H. Xue and X.-F. Jin, "Characters recognition of korean historical document base on data augmentation," in *2020 5th International Conference on Mechanical, Control and Computer Engineering (ICMCCE)*, pp. 2304–2308, 2020.

[22] T. Hayashi, K. Gyohten, H. Ohki, and T. Takami, "A study of data augmentation for handwritten character recognition using deep learning," in *2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, pp. 552–557, 2018.

[23] D. Varghese and A. Tamaddoni-Nezhad, *One-Shot Rule Learning for Challenging Character Recognition*. 07 2020.

[24] Y. Zhang, S. Liang, S. Nie, W. Liu, and S. Peng, "Robust offline handwritten character recognition through exploring writer-independent features under the guidance of printed data," *Pattern Recognition Letters*, vol. 106, pp. 20–26, 2018.

[25] N. Shaffi and F. Hajamohideen, "Few-shot learning for tamil handwritten character recognition using deep siamese convolutional neural network," in *Applied Intelligence and Informatics* (M. Mahmud, M. S. Kaiser, N. Kasabov, K. Iftekharuddin, and N. Zhong, eds.), (Cham), pp. 204–215, Springer International Publishing, 2021.

[26] N. Dlamini and T. L. van Zyl, "Author identification from handwritten characters using siamese cnn," in *2019 International Multidisciplinary Information Technology and Engineering Conference (IMITEC)*, pp. 1–6, 2019.

[27] Z. Li, M. Jin, Q. Wu, and H. Lu, "Deep template matching for offline handwritten chinese character recognition," *CoRR*, vol. abs/1811.06347, 2018.

[28] A. Fedele, R. Guidotti, and D. Pedreschi, "Explaining siamese networks in few-shot learning," *Machine Learning*, pp. 1–38, 2024.

[29] Z. Zheng, H. Wu, L. Lv, H. Ye, C. Zhang, and G. Yu, "Iccl: Independent and correlative correspondence learning for few-shot image classification," *Knowledge-Based Systems*, vol. 266, p. 110412, 2023.

[30] G. R. Koch, "Siamese neural networks for one-shot image recognition," 2015.

[31] I.-A. Lungu, Y. Hu, and S.-C. Liu, "Multi-resolution siamese networks for one-shot learning," *2020 2nd IEEE International Conference on Artificial Intelligence Circuits and Systems (AICAS)*, pp. 183–187, 2020.

[32] D. Zhang, J. Lv, Z. Cheng, Y. Bai, and Y. Cao, "Siamese network combined with attention mechanism for object tracking," *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. XLIII-B2-2020, pp. 1315–1322, 2020.

[33] G. Brauwers and F. Frasincar, "A general survey on attention mechanisms in deep learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, p. 3279–3298, Apr. 2023.

[34] S. P. Moustakidis and P. Karlsson, "A novel feature extraction methodology using siamese convolutional neural networks for intrusion detection," *Cybersecurity*, vol. 3, pp. 1–13, 2020.

[35] P. Ranjan and A. Girdhar, "Deep siamese network with handcrafted feature extraction for hyperspectral image classification," *Multimedia Tools and Applications*, vol. 83, no. 1, pp. 2501–2526, 2024.

[36] A. Hajavi and A. Etemad, "Siamese capsule network for end-to-end speaker recognition in the wild," in *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 7203–7207, 2021.

[37] M. Loster, I. Koumarelas, and F. Naumann, "Knowledge transfer for entity resolution with siamese neural networks," *J. Data and Information Quality*, vol. 13, jan 2021.

[38] J. Shen, Y. Liu, X. Dong, X. Lu, F. S. Khan, and S. Hoi, "Distilled siamese networks for visual tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 12, pp. 8896–8909, 2022.

[39] K. Sharma, C. Gong, R. Liu, N. Zhou, J. Luo, and D. Kumar Jain, "Smart memory storage solution and elderly oriented smart equipment design under deep learning," *Computational Intelligence and Neuroscience*, vol. 2022, p. 6448302, 2022.

[40] M. Leyva-Vallina, N. Strisciuglio, and N. Petkov, "Generalized contrastive optimization of siamese networks for place recognition," *CoRR*, vol. abs/2103.06638, 2021.

[41] Q. Wu, L. Liu, and S. Xue, "Global update guided federated learning," in *2022 41st Chinese Control Conference (CCC)*, pp. 2434–2439, 2022.

[42] T. Müller, G. Pérez-Torró, and M. Franco-Salvador, "Few-shot learning with siamese networks and label tuning," 2022.

[43] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[44] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4700–4708, 2017.

[45] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *International conference on machine learning*, pp. 6105–6114, PMLR, 2019.

ANIRUDHA GHOSH received his B.Sc. degree from the University of North Bengal in 2017 and his M.Sc. degree from the University of Gour Banga in 2019. He is currently pursuing a full-time Ph.D. in Deep Meta-Learning at Visva-Bharati, supported by a prestigious DST-INSPIRE fellowship awarded by the Government of India. His research interests include artificial intelligence for security, multimedia forensics, few-shot learning, computer vision, and data generation.

DEBADITYA BARMAN has obtained B.E. (2008), M.E. (2012), and Ph.D. (2021) in Computer Science and Engineering from Jadavpur University, India. At present, he is working as an Assistant Professor in the Department of Computer and System Sciences, Visva-Bharati, India. He is a senior member of IEEE. His main areas of research interest are machine learning, graph mining, and social network analysis.

ABU SUFIAN completed his Ph.D. in Computer Science from Visva-Bharati, India. He joined the National Research Council of Italy - Institute of Applied Sciences and Intelligent Systems (ISASI) in 2024 as a postdoctoral researcher, contributing to the consortium project "Future Artificial Intelligence Research – FAIR." He has a 13-year academic career, including nearly 9 years as an Assistant Professor at the University of Gour Banga, India. Sufian previously headed the Department of Computer Science at the University of Gour Banga, and contributed to academic planning and curriculum development. An active member of IEEE and the Computer Society of India, he has organized international events like the ICCV 2023'ACVR workshop, and the upcoming ECCV 2024' ACVR workshop.

IBRAHIM A. HAMEED holds a PhD in AI from Korea University, South Korea, and a PhD in field robotics from Aarhus University, Denmark. He is currently a Professor at the Norwegian University of Science and Technology (NTNU) and the Norwegian University of Life Science (NMBU), Norway. Hameed is an IEEE senior member, elected chair of the IEEE Computational Intelligence Society (CIS) Norway section, and founder and head of the Social Robots Lab in Ålesund. He has participated in several research projects funded by both Norwegian and European research councils. His current research interests include Artificial Intelligence, Machine Learning, Optimization, and Robotics.