Performance Analysis of Few-Shot Learning Approaches for Bangla Handwritten Character and Digit Recognition

Mehedi Ahamed, Radib Bin Kabir, Tawsif Tashwar Dipto,
Mueeze Al Mushabbir, Sabbir Ahmed and Md.Hasanul Kabir

December 7, 2024

2024 6th International Conference on Sustainable Technologies for Industry 5.0 (STI)
14-15 December, Dhaka

Performance Analysis of Few-Shot Learning Approaches for Bangla Handwritten Character and Digit Recognition

Mehedi Ahamed, Radib Bin Kabir, Tawsif Tashwar Dipto,
Mueeze Al Mushabbir, Sabbir Ahmed, and Md. Hasanul Kabir
Department of Computer Science and Engineering, Islamic University of Technology, Gazipur, Bangladesh
Email: {mehediahamed, radib, tawsiftashwar, almushabbir, sabbirahmed, hasanul}@iut-dhaka.edu

***Abstract*—Few-shot learning (FSL) offers a promising solution for classification tasks with limited labeled examples, offering a valuable solution for languages with limited annotated samples. Traditional deep learning research has largely centered on optimizing performance using large-scale datasets, yet constructing extensive datasets for all languages is both labor-intensive and impractical. FSL offers a compelling alternative, achieving effective results with minimal data. In this connection, this study investigates the performance of FSL approaches in Bangla characters and numerals recognition with limited labeled data, demonstrating their applicability to scripts with intricate and complex structures where dataset scarcity is prevalent. Given the complexity of Bangla scripts, we posit that models capable of performing well on these characters will generalize effectively to languages of similar or lower structural complexity. We introduce SynergiProtoNet, a hybrid network designed to enhance the recognition accuracy of handwritten characters and digits. Our model combines advanced clustering methods with a robust embedding framework to capture fine-grained details and contextual subtleties, leveraging multi-level (high- and low-level) feature extraction within a prototypical learning framework. We rigorously benchmark SynergiProtoNet against several state-of-the-art few-shot learning models, including BD-CSPN, Prototypical Network, Relation Network, Matching Network, and SimpleShot, across diverse evaluation settings. Our experiments—– Monolingual Intra-Dataset Evaluation, Monolingual Inter-Dataset Evaluation, Cross-Lingual Transfer, and Split Digit Testing demonstrate that SynergiProtoNet consistently achieves superior performance, establishing a new benchmark in few-shot learning for handwritten character and digit recognition. The code is available on GitHub: https://github.com/MehediAhamed/SynergiProtoNet.**

***Index Terms*—Handwritten Character Recognition, Few-Shot Classification, Cross-Lingual Transfer, Prototypical Network, SynergiProtoNet**

I. Introduction

Automated handwritten character and digit recognition systems are integral to Optical Character Recognition (OCR) applications and document analysis [1], [2]. Bangla, ranked as the seventh most spoken language globally with over 270 million speakers, [3], [4], where the script consists of 50 characters (11 vowels and 39 consonants), numerous compound characters, and 10 numerals, is both intricate and challenging for automated recognition. The variability in handwriting styles [5] and the high similarity between certain characters further complicate the task of accurate recognition [6]–[8]. These challenges are particularly relevant in sectors such as education, banking, and government services, where reliable recognition is essential for digitizing documents, verifying identity records, and processing financial transactions.

Despite such solutions being relevant for all languages, their development is often hindered owing to the scarcity of large-scale datasets across different scripts. Although Deep Learning (DL) based solutions have achieved state-of-the-art performance in a wide variety of computer vision tasks owing to their powerful ability to learn complex and abstract features [9]–[12], most of the conventional CNN and DL-based models proposed for handwritten character and digit recognition [13]–[16] require large-scale labeled datasets, which are often unavailable for low-resource languages. Few-shot learning (FSL) has emerged as a promising technique to overcome these hurdles by enabling models to learn effectively from limited data, thus improving the accuracy and efficiency of recognition systems across industries [17]–[19]. By learning from a few examples, FSL mitigates the need for large datasets, accelerating the development of effective recognition systems for underrepresented languages.

Amongst the handful of recent works in handwritten character recognition using FSL models, Sahay and Coustaty [20] leveraged a technique for Urdu handwritten character recognition by addressing its bi-directional nature, while Samuel et al. [21] applied similar techniques to Amharic scripts, capturing its unique features. For Tamil script, Shaffi and Hajamohideen [22] combined CNNs with RNNs to enhance recognition rates. FSL has also shown promise in low-resource languages like Persian [23] and in tasks like signature verification [24], with novel models proving effective in few-shot scenarios [25]. These advancements highlight the potential of few-shot learning models in improving handwritten character recognition across various scripts, however, a thorough analysis of the recent state-of-the-art FSL-based models is yet to be investigated in the literature.

This study presents a critical performance analysis of FSL approaches for handwritten Bangla characters and digit recognition. We introduce SynergiProtoNet, a hybrid model that combines a CNN encoder and a ResNet18 backbone to achieve a rich and diverse feature representation. The CNN encoder captures low-level features such as edges and textures, while the ResNet18 component extracts high-level, complex features, resulting in a model that excels in few-shot learning tasks. SynergiProtoNet sets a new benchmark in this task by addressing the complexities of the Bengali script and leveraging FSL for superior recognition accuracy and efficiency.

The proposed solution reduces the requirement for extensive data collection, which is expensive and resource-demanding. Adoption of our model in industry applications can improve the performance of automatic information retrieval from handwritten documents, streamlining workflows and reducing manual data entry errors in sectors such as banking, insurance, and government services, hence contributing to the big picture of achieving sustainable technology for the modern era.

II. Methodology

In this study, we leveraged Few-Shot Learning (FSL) to tackle the challenge of recognizing Bangla handwritten characters and digits with limited data. FSL enables generalization from limited examples through the use of a 'support set'— a small labeled dataset that provides a few representative samples per class, and a 'query set'— which includes unlabeled examples for classification after the model has been exposed to the support set. A novel class refers to categories that the model encounters for the first time during testing.

A. Datasets

To ensure a thorough performance analysis of the FSL models under different evaluation methods, we have leveraged four datasets namely BanglaLekha-Isolated [26], CMATERdb 3.1.2 [27], Devanagari dataset [28], and NumtaDB [29].

BanglaLekha-Isolated is a comprehensive dataset containing approximately 110,000 images across 50 Bangla characters and 10 numerals, collected from a diverse pool of writers, capturing significant variations in handwriting styles. CMATERdb 3.1.2 provides 300 images per class and includes characters from Bangla, Devanagari, and Tamil scripts, offering a multi-script dataset for evaluating cross-lingual performance. NumtaDB is focused on Bengali handwritten digits, with contributions from over 2,700 writers, encompassing various transformations and conditions to capture real-world variability. The Devanagari dataset includes 36 characters and 10 numerals, comprising 78,200 images in the training set and 13,800 images in the test set.

B. Task Formulation

Our approach centers on 'episodic training', where each episode simulates a few-shot scenario, aligning training tasks closely with the conditions anticipated during testing. In this context, each 'task' comprises inputs and expected outputs that the model must learn to recognize. The query set contains 10 random samples from the dataset for our experiments, evaluated under 1-shot, 5-shot, and 10-shot scenarios. The training involved 500 tasks per epoch, while validation employed 100 tasks. Instead of batch training, we opted for episodic training to prepare the model for real-world few-shot tasks more effectively.

To rigorously evaluate the model's robustness, we conducted both monolingual and cross-lingual experiments. Cross-lingual learning involves training on the script of one language and testing on another, while monolingual learning involves both training and testing on the script of the same language. Our experimental framework consisted of four evaluation methods:

1) Monolingual Intra-Dataset Evaluation: This method involved training and testing on the same dataset but with different character classes. We trained on consonants (39 classes) and tested on vowels (11 classes) from the CMATERdb 3.1.2 dataset [27]. This approach evaluated the model's ability to generalize within the same dataset to unseen character types.

2) Monolingual Inter-Dataset Evaluation: This evaluation assessed the model's capability to generalize to a different dataset with similar but distinct data distributions. In this method, we trained the model on the BanglaLekha-Isolated dataset's consonants and numerals (49 classes) [26] and tested it on the CMATERdb 3.1.2 dataset's vowels (11 classes).

3) Cross-Lingual Transfer: To evaluate cross-script generalization, we trained on the Devanagari dataset (46 classes) [28] and tested on the Bengali CMATERdb 3.1.2 (50 classes). This method tested the model's ability to transfer knowledge learned from one script (Devanagari) to another (Bengali).

4) Split Digit Testing: This approach was designed to evaluate the model's generalization ability within numerals. For this method, we used the NumtaDB dataset [29], training on digits 0-5 (6 classes) and testing on digits 6-9 (4 classes).

This multi-faceted evaluation strategy enables a thorough assessment of our FSL model's versatility and robustness across varying conditions and data sources, underscoring its potential for practical applications in handwritten character recognition across diverse linguistic contexts.

C. Few Shot Learning Approaches

1) Matching Network: This network introduces an attention-based framework for FSL, enabling efficient comparison between query and support images [30]. Images are initially embedded into a feature space via a CNN to produce feature vectors, and an attention mechanism calculates similarity scores between the query and support images using a similarity metric, typically cosine similarity. Classification of the query image is performed by aggregating the support labels weighted by these similarity scores, allowing for effective label propagation across limited samples.

2) Relation Network: The Relation Network framework enables flexible few-shot classification by learning a deep, trainable distance metric for comparing query images to support samples [31]. The images are embedded into a feature space through a CNN, and embeddings from the support and query sets are concatenated before being processed by a relation module that outputs a relation score, representing similarity. In k-shot settings, support embeddings are aggregated to form a class-level feature map for each class, which is then compared to the query. Training employs mean squared error (MSE) loss,

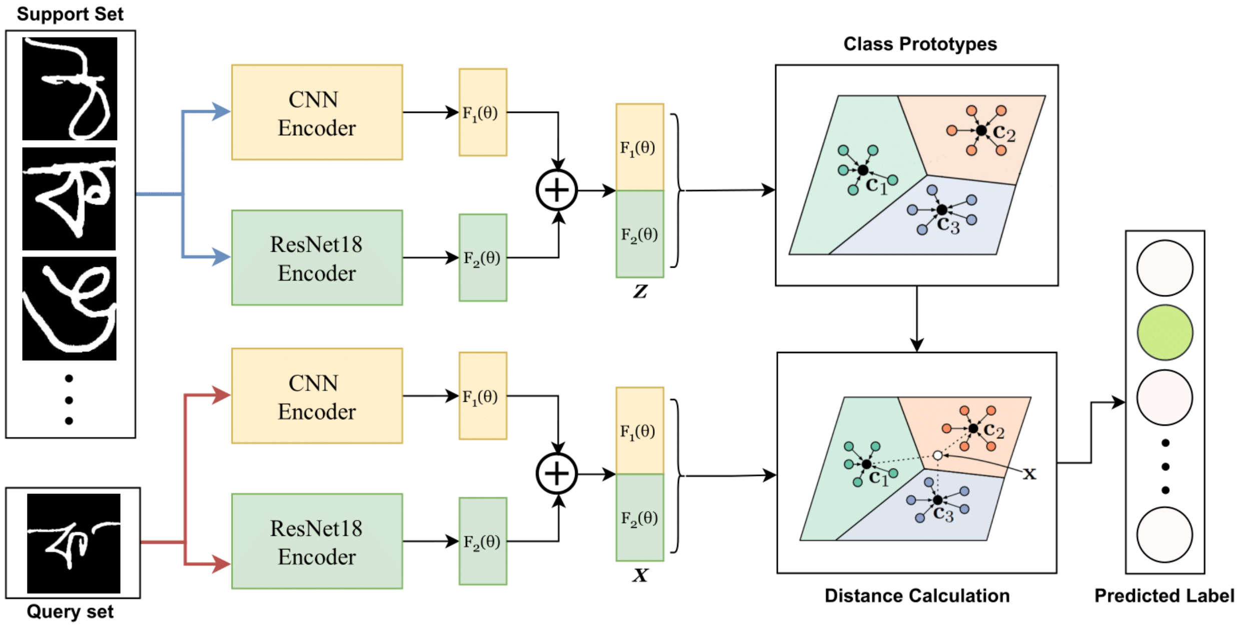

Fig. 1: **Overview of our proposed SynergiProtonet**. The architecture consists of a dual-encoder system, integrating CNN and ResNet18 encoders, to process both Support Set and Query Set samples. Each image is passed through both encoders, yielding feature vectors $F_1(\theta)$ from the CNN and $F_2(\theta)$ from the ResNet18. These feature vectors are concatenated to form a combined representation Z for each Support image, and a Query feature vector X for the Query image. Class prototypes C_i are calculated by averaging the combined feature vectors of each class in the Support Set. For classification, the Euclidean distance is computed between X and each prototype C_i, with a softmax function applied to the distances to yield class probabilities, allowing the model to classify the Query image based on its proximity to class prototypes in the feature space.

with matched pairs labeled as 1 and mismatched pairs as 0, allowing the model to optimize for accurate similarity learning. During inference, the query is classified based on the highest relation score among the support samples.

3) BD-CSPN (Bias Diminishing Cosine Similarity Prototypical Network): BD-CSPN advances FSL by refining class prototypes to mitigate intra- and inter-class biases, improving classification accuracy and robustness [32]. This approach begins by extracting features with a CNN and generating basic prototypes as the average of support set embeddings per class. To address intra-class bias, BD-CSPN incorporates high-confidence query samples into the support set via pseudo-labeling, recalculating prototypes as a weighted average. For cross-class bias, it shifts the query embeddings towards support set prototypes, refining classification through cosine similarity with the nearest adjusted prototype.

4) SimpleShot: The SimpleShot architecture is a lightweight approach to few-shot classification based on nearest-neighbor principles [33]. After feature extraction with a CNN, SimpleShot applies centering by subtracting the mean feature vector of base classes and normalizes embeddings to unit norm using L2 normalization. For classification, it employs a nearest-neighbor classifier with Euclidean distance. In one-shot settings, the query is classified based on the closest support sample, while in multi-shot cases, class centroids are derived from averaged support features, enhancing classification efficiency.

5) Prototypical Network: Prototypical Networks represent each class through a prototype computed as the mean embedding of the support set [34]. A CNN transforms images into high-dimensional vectors, with prototypes serving as centroids in embedding space. Classification is performed by calculating the Euclidean distance between the query embedding and each prototype and assigning the query to the nearest prototype. A softmax function transforms these distances into probabilities, while log-softmax loss penalizes misclassifications during training, allowing efficient learning from minimal data and rendering Prototypical Networks particularly well-suited to data-scarce environments.

D. SynergiProtoNet

Our proposed architecture, ‘SynergiProtoNet’ integrates both a CNN-based encoder and a ResNet18-based encoder, as shown in Fig. 1. It is designed to capture a rich spectrum of features for robust few-shot classification.

The CNN Encoder is designed to capture fine-grained, low-level features critical for differentiating subtle patterns in handwritten characters. The first two layers employ 3 × 3 convolutional layers with 64 filters, followed by Batch Normalization (momentum=1), ReLU activation, and 2 × 2 Max Pooling. These layers are optimized to detect foundational image elements, including edges, textures, and simple shapes. Layers 3 & 4 utilize additional 3 × 3 convolutional layers (64 filters, padding=1) with Batch Normalization and ReLU activation. These layers extend the receptive field, facilitating the extraction of intermediate-level features such as contours, corners, and detailed textural patterns.

The ResNet encoder utilizes a pre-trained ResNet18 model, focusing on extracting higher-level features by leveraging deep network architecture. Residual Connections mitigate the vanishing gradient problem, allowing the network to learn deeper, more abstract features without degradation. Deep Layers build increasingly complex feature hierarchies, extracting high-level features such as complex textures and object configurations essential for distinguishing between broader classes.

The combined encoder shown in Fig. 1 merges the outputs of the CNN and ResNet18 encoders to create a comprehensive feature representation that leverages both detailed local features and abstract global features. The outputs from both encoders are concatenated along the feature dimension, resulting in a combined feature vector that encapsulates a wide range of spatial information. This includes low-level features like edges, textures, corners, and fine patterns from the CNN Encoder; along with high-level features like complex textures and object configurations from the ResNet18 Encoder. This hybrid approach ensures that the feature representation is rich and diverse, capturing both intricate details and broader context, enhancing the model's ability to accurately classify few-shot learning tasks. The final output is flattened into a 1-dimensional vector, preserving detailed spatial information essential for distinguishing subtle differences between classes.

The few-shot classifier, implemented using Prototypical Networks, operates by computing class prototypes from the combined feature embeddings of the support set and classifying query samples based on their proximity to these prototypes. The embeddings are created by concatenating the outputs of the CNN and ResNet18 encoders along the feature dimension, resulting in a combined feature vector z that encapsulates both low-level and high-level features (Equation 1).

$$z = \text{concatenate}(f_{\text{CNN}}(x), f_{\text{ResNet}}(x), \text{dim} = 1) \tag{1}$$

where f_{CNN} and f_{ResNet} are the feature extraction functions of the CNN and ResNet18 encoders, respectively.

For each class c in the support set S, the prototype p_c is computed as the mean vector of the embedded support examples $\{z_i\}_{i=1}^{K}$, where K is the number of support examples per class, as shown in (Equation 2).

$$p_c = \frac{1}{K} \sum_{i=1}^{K} f_\phi(x_i^{(c)}) \tag{2}$$

To classify a query image x_q, the Euclidean distance between its embedding $f_\phi(x_q)$ and each class prototype p_c is computed (Equation 3).

$$d(z_q, p_c) = \|z_q - p_c\|_2 \tag{3}$$

Finally, a softmax function is applied over the negative distances to convert them into probabilities (Equation 4).

$$p_\phi(y = c \mid z_q) = \frac{\exp(-d(z_q, p_c))}{\sum_{c'=1}^{N_c} \exp(-d(z_q, p_{c'}))} \tag{4}$$

During training, the log-softmax loss is used to penalize the model when it fails to predict the correct class, driving the backpropagation process to minimize classification errors. This overall training process enables SynergiProtoNet to generalize effectively from limited examples, achieving high accuracy in challenging few-shot classification tasks.

E. Training and Evaluation Procedures

The model parameters were optimized using the SGD algorithm with a learning rate of 10^{-2}, momentum of 0.9, and weight decay of 5×10^{-4}. To ensure uniformity and compatibility, images were resized to $84 \times 84 \times 3$ to satisfy the input requirements of the ResNet18 backbone. A Multi-Step scheduler adjusted the learning rate at specified milestones, reducing it by a factor of 0.1 for fine-tuning. Cross-entropy loss was used to measure the discrepancy between predicted and true labels, making it suitable for multi-class classification tasks. The training process spanned 30 epochs, with episodic training involving 500 tasks for training and 100 tasks for validation in each epoch. For Bangla characters, FSL was performed using a 5-way classification setup, while for digits, a 3-way classification setup was used. During each episode, the support set images and labels were used to update the model's prototypes, which were then used to predict the classes of the query set images. Overlapping classes were avoided in all experiments to ensure fair evaluation.

III. Result Analysis

SynergiProtoNet, with its hybrid architecture, leverages a CNN for extracting lower-level local features and a pre-trained ResNet18 for capturing high-level global features. This fusion of detailed and abstract feature extraction facilitates comprehensive representation, enabling the model to perform consistently across diverse tasks and datasets.

1) Monolingual Intra-Dataset Performance: Table I reveals SynergiProtoNet's capacity on the CMATERdb 3.1.2 dataset, where it achieves the highest accuracy and F1 score across all shot scenarios. This impressive performance underscores the model's adeptness in handling the intricate and varied forms of handwriting that can exist within a single dataset, successfully distinguishing complex patterns and subtle distinctions between individual characters. SynergiProtoNet's architecture, meticulously designed to balance depth with precision, enables it to capture even the most nuanced variations in character formation, which are often missed by other models. While competing models struggle to maintain accuracy when faced with complex handwriting differences, SynergiProtoNet's deeper and more sophisticated structure empowers it to process and learn from these detailed features, consistently delivering high-precision recognition. This capability is particularly vital in real-world applications where script recognition needs to accommodate a high degree of variability without compromising on accuracy. By setting itself apart with such refined intra-dataset recognition, SynergiProtoNet not only demonstrates its robustness in controlled scenarios but also suggests promising

TABLE I: Monolingual Intra-Dataset Evaluation

Network	**1-shot**		**5-shot**		**10-shot**	
	Acc(%)	**F1**	**Acc(%)**	**F1**	**Acc(%)**	**F1**
Matching [30]	69.64	0.69	38.66	0.39	36.36	0.36
Simpleshot [33]	69.42	0.7	75.68	0.78	82.9	0.82
Relation [31]	77.10	0.76	85.58	0.87	83.2	0.83
BD-CSPN [32]	69.68	0.71	76.58	0.76	83.48	0.82
Prototypical [34]	74.48	0.74	87.88	0.87	87.56	0.87
SynergiProtoNet (ours)	**79.1**	**0.79**	**88.95**	**0.88**	**90.04**	**0.9**

TABLE II: Monolingual Inter-Dataset Evaluation

Network	**1-shot**		**5-shot**		**10-shot**	
	Acc(%)	**F1**	**Acc(%)**	**F1**	**Acc(%)**	**F1**
Matching [30]	48.6	0.52	41.8	0.39	26.9	0.26
Simpleshot [33]	51.34	0.55	63.54	0.63	64.94	0.66
Relation [31]	56.46	0.58	74.02	0.72	65.04	0.67
BD-CSPN [32]	50.84	0.5	60.3	0.6	69.2	0.7
Prototypical [34]	54.28	0.55	76.18	0.75	76.24	0.75
SynergiProtoNet (ours)	**59.02**	**0.59**	**77.68**	**0.78**	**81.36**	**0.83**

TABLE III: Cross-Lingual Performance Analysis

Network	**1-shot**		**5-shot**		**10-shot**	
	Acc(%)	**F1**	**Acc(%)**	**F1**	**Acc(%)**	**F1**
Matching [30]	52.84	0.52	26.68	0.27	37.58	0.38
Simpleshot [33]	42.92	0.46	56.14	0.53	55.10	0.54
BD-CSPN [32]	44.58	0.45	54.72	0.52	57.72	0.57
Prototypical [34]	53.64	0.55	76.74	0.77	79.48	0.79
Relation [31]	**61.12**	**0.61**	74.02	0.72	81.93	0.82
SynergiProtoNet (ours)	58.59	0.55	**76.84**	**0.77**	**82.12**	**0.82**

TABLE IV: Performance Analysis of Split Digit Testing

Network	**1-shot**		**5-shot**		**10-shot**	
	Acc(%)	**F1**	**Acc(%)**	**F1**	**Acc(%)**	**F1**
Matching [30]	**73.67**	**0.74**	33.1	0.29	32.17	0.33
BD-CSPN [32]	55.43	0.59	64.43	0.64	69.17	0.68
Simpleshot [33]	65.6	0.61	72.9	0.7	76.97	0.76
Relation [31]	64.97	0.63	79.83	0.8	81.37	0.83
Prototypical [34]	66.57	0.65	76.33	0.75	87.1	0.86
SynergiProtoNet (ours)	37.4	0.37	**83.73**	**0.83**	**88.3**	**0.87**

adaptability to real-world complexities in handwritten character recognition across different settings and styles.

2) Monolingual Inter-Dataset Performance: Table II demonstrates SynergiProtoNet's robustness when trained on one dataset containing Bangla handwritten scripts and evaluated on a different one, where the training and testing sets contained entirely distinct classes of the same language. The superior performance of SynergiProtoNet in this setting can be attributed to its enhanced encoder, which initially captures intricate, task-relevant features, subsequently refined by ResNet18's deep feature extraction. This layered feature capture enables better generalization across datasets, adapting to the inherent diversity of Bangla handwritten characters more effectively than other models. In contrast, models such as Prototypical Networks and Relation Networks, while performing reasonably well, lack sophisticated feature extraction capabilities, leading to slightly lower performance metrics. Similarly, the performance of Simpleshot, BD-CSPN and Matching Networks, still lags behind SynergiProtoNet, underscoring due to similar reasons. The rigorous testing through this evaluation highlights SynergiProtoNet's versatility.

3) Cross-Lingual Performance Analysis: Table III explores the model's performance in cross-lingual scenarios, where it was trained on Devanagari characters and tested on Bengali characters. Here, SynergiProtoNet demonstrated strong performance in the 5-shot and 10-shot scenarios, maintaining balanced accuracy across multiple shots. Although the Relation Network achieved marginally higher accuracy in the 1-shot scenario, it lagged behind SynergiProtoNet in multi-shot settings. This consistent performance across shots illustrates SynergiProtoNet's ability to manage the cross-lingual shift effectively, even as other models struggle to generalize across the language barrier. Prototypical Networks perform reasonably well but have slightly lower generalization capacity compared to ours. SimpleShot architecture, relying on nearest-neighbor classification, performed lowest overall due to its limited feature extraction depth, unable to capture complex, language-specific nuances. BD-CSPN and Matching Networks show lower performance, particularly in the 5-shot scenario for Matching Networks, indicating a potential overfitting issue where the models fail to generalize from the provided examples. While SynergiProtoNet's performance in 1-shot is slightly diminished due to its architectural complexity which limits its trainability with such limited data, its results were on par with the other models. Its steady performance across other shots showcases its robustness in cross-lingual scenarios.

4) Split Digit Testing: Table IV evaluated the model on NumtaDB, a dataset featuring diverse Bengali numerals across five different subsets, each varying in style and format (e.g., handwritten in pencil, pen, or generated digitally on contrasting backgrounds). Although the Matching Network performed well in the 1-shot setting, SynergiProtoNet demonstrated significantly better accuracy in the 5-shot and 10-shot scenarios. The diversity in NumtaDB's digit styles presented a challenge for many models, with SynergiProtoNet's hybrid architecture showing resilience and adaptability in capturing the underlying structures across varied numeral styles. The model's performance illustrates its robustness, effectively handling both simple and complex numeral presentations, while other methods struggled to generalize from the diverse data.

In summary, these evaluations reveal that SynergiProtoNet's hybrid architecture is well-suited for handling intricate details and abstract representations, achieving top performance across monolingual, cross-lingual, and complex numeral datasets. This versatility positions SynergiProtoNet as a powerful solution for few-shot learning tasks in complex, diverse handwriting recognition applications.

IV. Conclusion

This study provides a thorough performance analysis of several state-of-the-art few-shot learning models in Bangla handwritten character and digit recognition under diversified circumstances. Moreover, we introduce SynergiProtoNet which is specifically designed to tackle the complexities of Bangla handwritten scripts with limited data. By employing a hybrid encoder structure that combines CNN-based local feature extraction with high-level representations from ResNet18, SynergiProtoNet captures a rich spectrum of visual details and contextual patterns, demonstrating its effectiveness across varied monolingual and cross-lingual scenarios. Experimental results indicate that SynergiProtoNet not only achieves state-of-the-art accuracy within Bangla datasets but also adapts effectively to cross-dataset evaluations, highlighting its resilience to inter-dataset variability and class diversity in handwritten scripts. Future research could improve SynergiProtoNet to support compound and non-Latin characters, enhancing its versatility across various scripts. Further refinement of its hybrid architecture with better attention mechanisms or adaptive feature fusion may increase robustness in diverse handwriting scenarios. This positions SynergiProtoNet as an effective solution for industries, promoting sustainable technology by minimizing continuous data acquisition and retraining.

REFERENCES

[1] 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. 142 642–142 668, 2020.

[2] H. Singh and A. Sachan, "A proposed approach for character recognition using document analysis with ocr," in *2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS)*. IEEE, 2018, pp. 190–195.

[3] S. Aziz, N. H. Arif, S. Ahbab, S. Ahmed, T. Ahmed, and M. H. Kabir, "Improved speech emotion recognition in bengali language using deep learning," in *2023 26th International Conference on Computer and Information Technology (ICCIT)*, 2023, pp. 1–6.

[4] A. Yasmeen, F. I. Rahman, S. Ahmed, and M. H. Kabir, "Csvc-net: Code-switched voice command classification using deep cnn-lstm network," in *2021 Joint 10th International Conference on Informatics, Electronics & Vision (ICIEV) and 2021 5th International Conference on Imaging, Vision & Pattern Recognition (icIVPR)*, 2021, pp. 1–8.

[5] A. S. A. Rabby, S. Haque, M. S. Islam, S. Abujar, and S. Hossain, "Bornonet: Bangla handwritten characters recognition using convolutional neural network," *Procedia Computer Science*, vol. 143, pp. 528–535, 01 2018.

[6] A. B. M. Ashikur Rahman, M. B. Hasan, S. Ahmed, T. Ahmed, M. H. Ashmafee, M. R. Kabir, and M. H. Kabir, "Two decades of bengali handwritten digit recognition: A survey," *IEEE Access*, vol. 10, pp. 92 597–92 632, 2022.

[7] T. Ahmed, M. N. Raihan, R. Kushol, and M. S. Salekin, "A complete bangla optical character recognition system: An effective approach," in *2019 22nd International Conference on Computer and Information Technology (ICCIT)*, 2019, pp. 1–7.

[8] T. I. Aziz, A. S. Rubel, M. S. Salekin, and R. Kushol, "Bangla handwritten numeral character recognition using directional pattern," in *2017 20th International Conference of Computer and Information Technology (ICCIT)*, 2017, pp. 1–5.

[9] A. Khatun, M. S. Shahriar, M. H. Hasan, K. Das, S. Ahmed, and M. S. Islam, "A systematic review on the chronological development of bangla sign language recognition systems," in *2021 Joint 10th International Conference on Informatics, Electronics & Vision (ICIEV) and 2021 5th International Conference on Imaging, Vision & Pattern Recognition (icIVPR)*, 2021, pp. 1–9.

[10] S. Ahmed, M. B. Hasan, T. Ahmed, M. R. K. Sony, and M. H. Kabir, "Less is more: Lighter and faster deep neural architecture for tomato leaf disease classification," *IEEE Access*, vol. 10, pp. 68 868–68 884, 2022.

[11] R. M. Alamgir, A. A. Shuvro, M. Al Mushabbir, M. A. Raiyan, N. J. Rani, M. M. Rahman, M. H. Kabir, and S. Ahmed, "Performance analysis of yolo-based architectures for vehicle detection from traffic images in bangladesh," in *2022 25th International Conference on Computer and Information Technology (ICCIT)*, 2022, pp. 982–987.

[12] A. N. Ashik, M. S. H. Shanto, R. H. Khan, M. H. Kabir, and S. Ahmed, "Recognizing bangladeshi traffic signs in the wild," in *2022 25th International Conference on Computer and Information Technology (ICCIT)*, 2022, pp. 1004–1009.

[13] S. Maity, A. Dey, A. Chowdhury, and A. Banerjee, "Handwritten bengali character recognition using deep convolution neural network," in *Machine Learning, Image Processing, Network Security and Data Sciences*. Singapore: Springer Singapore, 2020, pp. 84–92.

[14] C. Saha and M. M. Rahman, "Banglanet: Bangla handwritten character recognition using ensembling of convolutional neural network," *arXiv*, Jan. 2024.

[15] A. S. A. Rabby, S. Haque, M. S. Islam, S. Abujar, and S. Hossain, "Bornonet: Bangla handwritten characters recognition using convolutional neural network," *Procedia Computer Science*, vol. 143, pp. 528–535, 01 2018.

[16] M. Kamal, F. Shaiara, C. M. Abdullah, S. Ahmed, T. Ahmed, and M. H. Kabir, "Huruf: An application for arabic handwritten character recognition using deep learning," in *25th International Conference on Computer and Information Technology (ICCIT)*, 2022, pp. 1131–1136.

[17] Y. Wang, Q. Yao, J. T. Kwok, and L. M. Ni, "Generalizing from a few examples: A survey on few-shot learning," *ACM Comput. Surv.*, vol. 53, no. 3, jun 2020.

[18] S. Ahmed, "Classification of plant disease from leaf images using few-shot learning," MSc Thesis, Department of Computer Science and Engineering (CSE), Islamic University of Technology, 2022.

[19] M. A. Rahman, N. I. Asad, M. M. H. Omi, M. B. Hasan, S. Ahmed, and M. H. Kabir, "Fused-net: Enhancing few-shot traffic sign detection with unfrozen parameters, pseudo-support sets, embedding normalization, and domain adaptation," *arXiv*, 2024.

[20] R. Sahay and M. Coustaty, "An enhanced prototypical network architecture for few-shot handwritten urdu character recognition," *IEEE Access*, vol. PP, pp. 1–1, 01 2023.

[21] M. Samuel, L. Schmidt-Thieme, D. P. Sharma, A. Sinamo, and A. Bruck, "Offline handwritten amharic character recognition using few-shot learning," in *Pan-African Conference on Artificial Intelligence*. Springer Nature Switzerland, 2023, pp. 233–244.

[22] N. Shaffi and F. Hajamohideen, "Few-shot learning for tamil handwritten character recognition using deep siamese convolutional neural network," in *Applied Intelligence and Informatics*. Cham: Springer International Publishing, 2021, pp. 204–215.

[23] A. Hajebrahimi, M. E. Santoso, M. Kovacs, and V. V. Kryssanov, "Few-shot learning for character recognition in persian historical documents," in *Machine Learning, Optimization, and Data Science*. Cham: Springer Nature Switzerland, 2024, pp. 259–273.

[24] P. Majumder, A. Joaa, E. R. Rhythm, M. H. K. Mehedi, and A. A. Rasel, "Siamese-transformer network for offline handwritten signature verification using few-shot," 12 2023, pp. 1–6.

[25] N. Elaraby, S. Barakat, and A. Rezk, "A novel siamese network for few/zero-shot handwritten character recognition tasks," *Computers, Materials and Continua*, vol. 74, pp. 1837–1854, 08 2022.

[26] M. Biswas, R. Islam, G. Shom, M. Shopon, N. Mohammed, S. Momen, and A. Abedin, "Banglalekha-isolated: A multi-purpose comprehensive dataset of handwritten bangla isolated characters," *Data in Brief*, vol. 12, pp. 103–107, 2017.

[27] M. M. Rahman, "Bangla handwritten character db cmaterdb 3.1.2," 2023, [accessed 24-August-2024]. [Online]. Available: https://www.kaggle.com/datasets/mostafiz53/basicfinal

[28] S. Sinha, "Hindi character recognition," Sep. 2021, [accessed 4-August-2024]. [Online]. Available: https://www.kaggle.com/datasets/suvooo/hindi-character-recognition

[29] S. Alam, T. Reasat, R. M. Doha, and A. I. Humayun, "Numtadb - assembled bengali handwritten digits," *arXiv*, 2018.

[30] O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, and D. Wierstra, "Matching networks for one shot learning," *Advances in neural information processing systems*, vol. 29, 2016.

[31] F. Sung, Y. Yang, L. Zhang, T. Xiang, P. H. S. Torr, and T. M. Hospedales, "Learning to compare: Relation network for few-shot learning," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[32] J. Liu, L. Song, and Y. Qin, "Prototype rectification for few-shot learning," in *Computer Vision – ECCV 2020*. Springer International Publishing, 2020, pp. 741–756.

[33] Y. Wang, W.-L. Chao, K. Q. Weinberger, and L. van der Maaten, "Simpleshot: Revisiting nearest-neighbor classification for few-shot learning," *arXiv*, 2019.

[34] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," in *Advances in Neural Information Processing Systems*, vol. 30. Curran Associates, Inc., 2017.