IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 13 March 2023, accepted 29 March 2023, date of publication 31 March 2023, date of current version 6 April 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3263721

RESEARCH ARTICLE

An Enhanced Prototypical Network Architecture for Few-Shot Handwritten Urdu Character Recognition

RAJAT SAHAY[1] AND MICKAËL COUSTATY[2], (Member, IEEE)
[1]Vellore Institute of Technology, Vellore 632014, India
[2]Laboratoire Informatique, Image et Interaction, La Rochelle Université, 17042 La Rochelle, France
Corresponding author: Mickaël Coustaty (mickael.coustaty@univ-lr.fr)

ABSTRACT Few shot models have started to gain a lot of popularity in the past few years. This is mostly because these models grant the ability to structure the representation space (classes) using a very less amount of examples for each class. Such models are usually trained on a wide range of different classes and their examples, which allows them to form and learn a decision-based metric in the process. Non-Latin languages, especially languages such as Urdu, have a bi-linear direction of writing and are context-sensitive in nature, and are hard to recognize. Also, unlike traditional English, there is a very small amount of clean, collated, and usable data that is available for the Urdu language. In this paper, we explore a prototypical network for k-shot classification on handwritten Urdu characters. The prototypical network learns the Euclidean embeddings of the provided images and uses clusters to classify newer examples. Our improved method is able to outperform other methods of few-shot learning and is able to accurately classify both Urdu characters as well as numerals using a minimal number of examples. After comprehensive qualitative and quantitative evaluation and comparison of our proposed approach with other methods to classify handwritten text in few-shot settings, we found out that our proposed approach was typically able to beat other methods by a margin of $1\% - 2\%$ while relying on a small training set.

INDEX TERMS Few-shot learning, prototypical network, Urdu handwritten recognition.

I. INTRODUCTION

Urdu is the national language of Pakistan. It is widely spoken and understood as a second language by a majority of people of Pakistan [32], and Urdu handwriting is used for official assignments and communications in all the organizations. Urdu has also been observed to be spoken in other regions of South-East Asia, including India, Bangladesh, Afghanistan etc. Previous works [20], [22], [36] have shown that there is a sizeable gap in the amount of data that has been accumulated for Urdu when compared to the data that has been accumulated for the recognition of Latin-based languages (English, German etc).

Although fields like OCR have witnessed a significant improvement with advancements in vision technology, due to reasons mentioned above combined with the complexity of typesetting non-Latin languages, the OCR for Urdu is still in its initial stages of infancy [3], [47]. The problems affecting OCR are compounded often due to incorrect acquisition and segmentation processes during the annotation and recognition processes. The main motivation of this work is to imitate human reading ability, with human accuracy and a higher speed, even for data-sparse languages like Urdu.

Deep Learning has been gaining popularity in the past decade, and the continuous research conducted has only advanced the speed of progress. However, most of the deep learning methods that are developed are usually observed to perform better in settings with abundance of data. The field of computer vision and pattern recognition are no different and suffer from the same problem. Most methods proposed in this field usually rely on being iteratively trained with lots of data in a supervised manner. The problem of handwritten

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

text recognition has been one of the oldest and most widely explored problems in this field [6]. However, in some cases, like the one this paper explores, the previously proposed solutions to these problems are somewhat ineffective because they are unable to adapt to new or unseen data quickly.

Since there is a shortage of the amount of handwritten data accumulated for the recognition of Urdu characters, this paper proposes a few-shot learning based system that is able to recognize individual Urdu characters and digits from an extremely limited amount of given examples. We test our improved prototypical network based approach against other methods and are able to achieve better accuracy and outperform other methods. We note an average accuracy of 92% on characters in a 5-way-5-shot setting over multiple datasets, while showing an average classification accuracy of 90% for classifying numerals in a 5-way-5-shot settings. To assess relevance and generalization capabilities of our proposed approach, we tested our solution on two different datasets. These datasets are the one created by Husnain et al. [18], and the UHaT dataset [5]. We show in Table 1 and Table 2 that our proposed method was able to consistently outperform previous few-shot solutions for recognition tasks. The next section presents the related works (section II), our proposed approach (section III) and its evaluation (section IV).

II. RELATED WORKS

A. URDU HANDWRITTEN CHARACTER RECOGNITION

Recognizing handwritten text at the character level has been one of the first and foremost tasks that researchers have hoped to tackle using computer vision methods [6]. Recognition of handwritten characters was first put to use to recognize handwritten, and machine-printed text for the interpretation of postal addresses [42]. This is extraordinarily challenging since handwritten characters present large amounts of variations, even when written by the same person. Postal addresses, on the other hand, were written by many different people, each presenting a different style of writing. Previous studies and research [40] show that the handwriting of an individual is a direct consequence of various different factors, which include but are not limited to age, current mood, and immediate surroundings.

In the past few years, there have been sizeable developments in the field of handwritten character recognition [24], [30]. However, most of the research in this field deals with the recognition of the 26 alphabets and 10 digits of the Latin languages. Urdu is one of the most commonly used languages in countries from South Asia (like Pakistan and Afghanistan). Urdu is written right-to left in an extension of the Persian alphabet, which is itself an extension of the Arabic alphabet. Urdu is associated with the Nastaīq style of Persian calligraphy, which is notoriously difficult to typeset (and then commonly handwritten). This difficulty leads to the non-availability of properly collated and annotated data, which is the primary reason for such less research in handwritten Urdu character recognition. Similar challenges also exist for typesetting Persian and Arabic characters, which has prompted an increase in similar research in innovative ways for character recognition for these languages as well [16], [38], [43]. This paper specifically focuses on the recognition of handwritten characters in the Urdu language. While the literature on generic handwritten text recognition is available, non-Latin languages like Urdu have inhibited research due to the lack of availability of data. Moreover, issues like context-sensitivity and non-standard shapes of characters make it a harder task compared to simple handwritten text recognition.

Many previous research works in the handwritten recognition field show the use of Artificial Neural Networks (ANNs) [44]. The problem with the development of a generic character recognition system is that it cannot be generalized to support all types of languages due to many variations between Latin and Asian languages. The first distinguishing feature of the Urdu script is that it is written and read from right-to-left, unlike traditional Latin languages. To overcome the problem of non-generalization of different languages, a novel approach was proposed [4] which mapped the character set of any language to a feature space representing them by using geometrical strokes that formed the characters. This allowed the ANN to be trained only once. It was able to achieve a success rate of approximately 75%-80%.

One other way in which Urdu differs from English and other traditional Latin languages is its intrinsic variability. Urdu script is written from right-to-left, while the numerals are written from left-to-right. This is why Urdu is considered as a *bidirectional* language. It consists of 38 alphabets and 10 numerals as displayed in Figure 1. Due to a large character set, and considering the fact that the Urdu script is context-sensitive, which means that the characters have different shapes and sizes based on their position in the word, the recognition of Urdu characters becomes increasingly difficult in contrast to English. Also, there are a lot of characters in Urdu that, in visual terms, look quite similar to each other. Keeping this in mind, authors of [15] grouped the Urdu characters on the basis of the number of strokes it took to construct the character. The novel features from each group were extracted and given as input to classifiers with different neural architectures. These included a Probabilistic Neural Network (PNN) and a Backpropogation-based Neural Network (BNN). The result showed that the PNN was able to outperform other classifiers. Based on experiments done in the paper, the PNN was able to outperform other classifier architectures by a margin of 6%. Previous studies done for lone character recognition [39] were able to sufficiently recognize handwritten Urdu characters using a feature vector that it built by the analysis of the strokes that were used to construct the character. There have been more recent works that tackle the problem of Urdu Character Recognition. For instance, Ahmed et al. [2] presented the offline-UNHD dataset for Urdu characters. Experiments were performed using a 1-dimensional Bi-Directional LSTM to classify individual character labels. Other works such as [35]

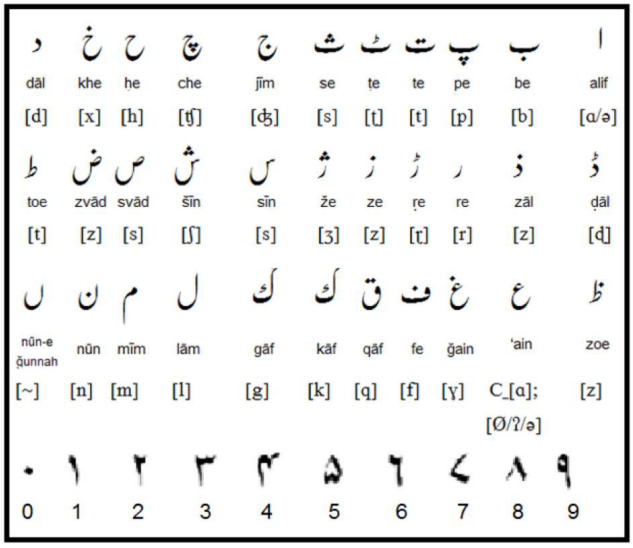


FIGURE 1. Urdu character set with phonemes and Urdu numerals with their corresponding Roman equivalences (from [18]).

use Transformer architectures with a CNN backbone for unconstrained character classification.

The large variability of Urdu characters and numerals representation is a challenge for deep-learning models as they need a large set of annotated data to learn their representation. To the best of our knowledge, only two public properly collated datasets could be used to train a model with a limited number of samples per class. The dataset used by Husnain et al. [18] proposed around 500 samples per class, while the Urdu Handwritten Text (UHaT) [5] dataset proposes around 700 samples per class. The lack of properly structured and annotated data has always been a major hindrance for these types of tasks. Most recent methods [2], [28], [35] that involve the use of data-intensive and hungry architectures like LSTMs and Transformers usually are unable to perform when provided with sparse datasets. With continually changing styles and patterns, the Urdu language provides just that in the real world. In order to tackle the large variability of Urdu language and to learn from a limited number of examples with supervised information, some new machine learning paradigm, called Few-Shot Learning, have been proposed in the middle on 2000's and widely studied since that time [50]. We propose to study them in our context.

B. FEW SHOT LEARNING

Deep learning methods have been observed to perform extremely well in settings and environments where there is an abundance of data available. Most conventional neural network architectures are designed in such a way that the influence of a particular example within the training set would be extremely small in terms of bringing about a change in the final results. These networks are designed to capture and learn the general features that are displayed throughout the dataset [37], which usually has a varied amount of examples.

There are however some areas that require fast adaption to new kinds of data, with a minimal number of given examples. The concept of one-shot, few-shot or k-shot learning is motivated by the general idea of how humans are able to learn a new class of objects with just a few given examples. These types of learning methods require the network to classify unseen classes of data using single, a few and k examples respectively.

The first working example of a model using a few-shot learning based approach was given by Vinyals et al. [45] in the form of *Matching Networks*. The training for this type of network was performed on sets of support examples for a subset of possible classes, and a query example belonging to one of the classes. Each training step was the combination of training the network on support classes and query classes and was referred to as an episode. These episodes were generated for a wide range of classes. The number of support examples would be varied based on the type of learning that the network was used to achieve. Each episode was used to essentially mimic the task of classifying data into classes with a minimal amount of examples by subsampling the number of classes as well as the data points in each of them. It has been demonstrated that this approach helped improve the overall performance and made the entire training problem more faithful to the testing environment. It also enabled the network to better generalize results to unseen examples. It would be given k examples for k-shot learning and a single example for one-shot learning. Reference [33] was able to take the idea of training a network with episodes a step ahead and combined it with an LSTM [17]. The authors proposed an approach that could be considered as a meta-learning approach to few-shot learning.

Non-parametric models, like k-nearest neighbours (kNN) can also be considered as an ideal candidate for few-shot classification. This is because they allow inclusion of previously unseen classes. They use distance in the space of inputs as a metric to define class boundaries, which prevents a high accuracy during classification. However, it has been seen that combinations of parametric and non-parametric methods have yielded the best results [23], [41].

In this paper, we explore *Prototypical Networks*, which do not suffer from severe overfitting and according to our knowledge are able to achieve state of the art performance on the *Omnilogot* dataset [41]. Furthermore, with the scarcity of annotated data for Urdu characters, we found that these networks consistently outperformed other few-shot learning approaches.

We were also able to improve the performance of a vanilla Prototypical Network on both the datasets of Urdu characters and numerals by adding modules for regularizing individual weights given to initial prototypes and balancing the distances between the clusters that are formed in the embedding vector space.

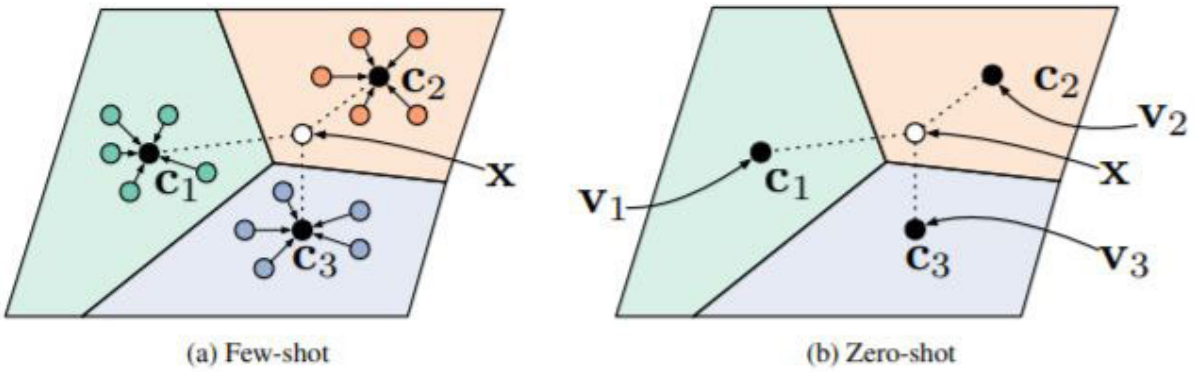


FIGURE 2. State of Prototypical Networks in Few Shot and Zero Shot Scenarios. Embedded query points are classified using the mean over distances to individual class prototypes (from [41]).

C. PROTOTYPICAL NETWORKS

Prototypical Networks were introduced by Snell et al. [41] for both few-shot as well as zero-shot settings. The authors of the paper drew connections to Matching Networks [45], which were used in one-shot learning problems, and built upon the underlying distance function. They related Prototypical Networks to clustering in order to justify the use of class means as prototypes when the distances are computed with a Bregman divergence, such as Squared Euclidian Distance.

The objective of a Prototypical Network is to learn the metric on the embedding space which is a representation of the similarity in the form of a distance metric (which, previous research [23], [45] has shown can be L2 distance or *cosine* distance). This method is easily able to adapt to general k-shot by using the mean of the k samples as the center of the cluster. They do not classify the image directly, but learn the mapping of an image in metric space. The encoder of the network maps the images belonging to the same class close to each other, while images that belong to entirely different classes are placed apart from each other by a considerable distance. Whenever a new example is given, the network confirms the nearest cluster to the encoding of the image in the metric space, and classifies the image accordingly.

They differ from Matching Networks as the latter produces a weighted nearest neighbour given the support set while the former uses squared Euclidian Distance to give a linear classifier as the output. In the case of a one-shot setting, where there is only one data-point available per class, both networks become equivalent. Vinyals et al. [45] and Ravi and Larochelle [33] both used the cosine distance as a metric to compute distance. However, the authors of [41] found that the squared Euclidian distance was better and improved results for both Prototypical as well as Matching Networks. Based on this study, we propose in the next of this paper, to develop our proposed approach using a prototypical network for Urdu handwritten recognition.

III. PROPOSED APPROACH

The approach that we followed was majorly derived from classical recognition problems in pattern recognition. We first preprocessed the images in the dataset to remove any unnecessary noise, resize them to a standard size, keeping the aspect ratio fixed. We also rotated them in multiples of 90 degrees to generate more classes. These images were then passed through a self-supervised encoder to obtain a generic representation of the features of the images. Finally, we use an improved Prototypical Network model (described in Sections 3.3, 3.4 and 3.5) to structure the representation space which we used to classify unseen data with a minimal number of examples.

Figure 3 gives a allegorical overview on the step-by-step process taken by our approach to classify handwritten Urdu text.

A. PREPROCESSING

As in all pattern recognition processes, data need to be cleaned to remove noises and irrelevant features. To this end, the images of the dataset are sent through basic preprocessing steps to prepare the individual images to be encoded. First, the images are processed to remove noise using methods that were used in noteworthy work conducted by Bieniecki et al. [8]. All the images are grayscaled and resized to 28×28 pixels (if there were any images that were not of this size) while keeping the aspect ratio fixed at all times. We also (similar to the authors of [41]) rotated the images by angles of 90°, 180° and 270° in order to augment the variability of images and gain in genericity. This last step was also done keeping in mind real-world uses of Urdu OCR applications. By using data augmentation, our model would be able to make more robust predictions when given 'dirty' examples, which are usually Urdu characters captured in different orientations. For our model to be deployed in practical scenarios, we deem it equally necessary that the confidence of our predictions our not altered by simply a change of the orientation of a provided character.

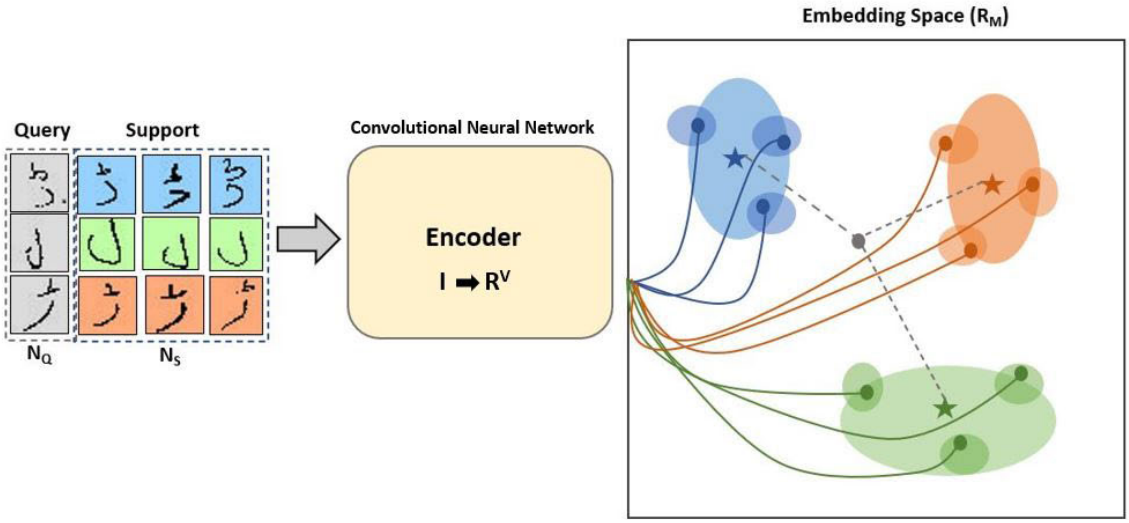


FIGURE 3. An overview of our proposed approach. The encoder (the CNN) converts an image to a vector ($\vec{x} \in \mathbf{R}^V$). Then, the prototypical network maps each image $\vec{x}$ into the embedding space R^M (dark circles in the figure) while the support images are used to define the prototypes c_k (stars in the figure). The distance between prototypes and encoded query images are used to classify the images.

B. ENCODER

Once data are cleaned, we propose to encode the given input images so that they can be mapped across a metric space for later classification. For this second step, we use a multi-layer convolutional neural network as an encoder network to convert the given image into a 64 dimensional point in the metric space. For practical purposes, we used 5 convolutional blocks. Each block consists of a two-dimensional, 3×3 convolutional layer, followed by batch normalization and a *ReLU* activation function [1]. This is followed by a 2×2 Max-Pooling layer. After the 5 blocks, the final result is returned as the flattened form of the output. When applied to the 28×28 sized images, it leads to a 64-dimensional output embedding. A schematic overview of this encoder is proposed in figure 4.

For a vanilla prototypical network, the encoder can be defined as a function

$$encoder(W) : I \in \mathbf{R}^{H*W*C} \rightarrow \vec{x} \in \mathbf{R}^V \tag{1}$$

where, I is the input image provided to the encoder, $\vec{x}$ is the embedded vector, H is the height of the image, W is the width of the image and C is the number of channels. V is the embedding dimension of the vector space.

The encoder network that we proposed was taken from the original paper which explored Prototypical Networks [41]. We explored other architectures for an encoder network by changing the architecture of the CNN, but we found that the original network outperformed the others. We used the same encoding network for embedding both the support as well as the query points.

For the loss function of the encoder, we decided to use the Negative Log Likelihood [52] loss function, because it is robust enough for the encoder to learn representations efficiently.

C. k-SHOT LEARNING USING PROTOTYPICAL NETWORK

The last step or our architecture is related to the representation space structuring. Few-shot learning environment are usually provided with a set of labelled examples, known as the *support set* (S). Let K be the set of classes (in our case, the set of all possible Urdu characters), the support set would have N image-label pairs in it, as $S = \{x_i, y_j\} | (1 <= i <= N, 1 <= j <= K)$, where each x corresponds to the vector obtained in the encoding step and each y belongs to the target space (ranging in $1, \ldots K$). The x belongs to the V-dimensional vector space (R^V) and S_k denotes the set of examples labelled with k^{th} class.

We used the original implementation of a Prototypical Network (pictorially denoted in Figure 2) with some added modules to obtain an M-dimensional vector space R^M allowing to represent each class(denoted (c_k)).Since the encoder reduces the original images into an R^V space, we map these encodings to an R^M space to reduce linearity and avoid sparsity in the embeddings that are going to be passed into the prototypical architecture. This is achieved through a function f_τ with learnable parameters τ. The function f_τ maps from the feature vector space R^V to the M-dimensional vector space, R^M. Each representation (c_k) is the value of the mean of all vectors of the embedded points in class k. It can be defined as:

$$c_k = \frac{1}{| S_k |} \sum_{x_i, y_i} f_\tau(x_i) \tag{2}$$

The aim of our prototypical network is to train a classifier, *i.e.* set the centroids of classes. This classifier is then able to generalize to new and previously unseen classes (*query set*)

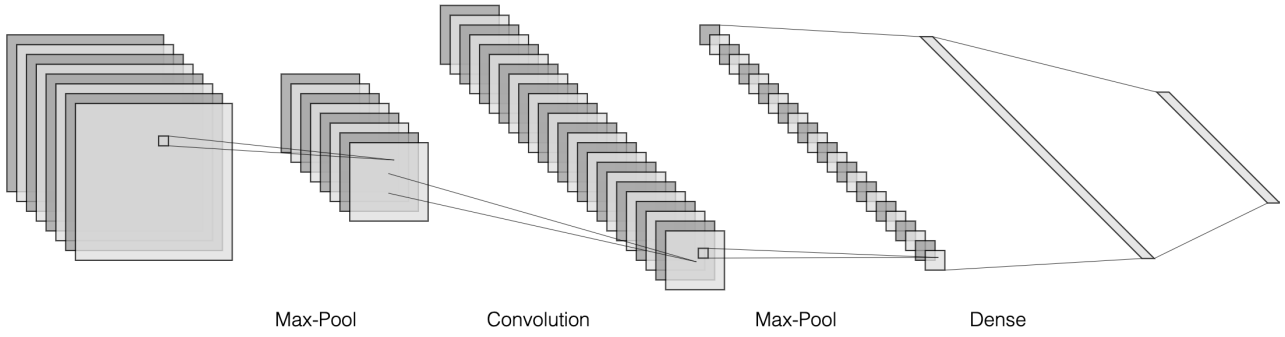


FIGURE 4. Structure of the multi-layered convolutional neural network that is used to encode the features of Urdu characters and numerals into a 64-dimensional vector space: R^V.

that were previously unavailable during the training. It also required an extremely small number of examples of each of the new classes (*the support set*). We randomly selected C classes from the input data. From each class, we chose a support set and a query set. We passed all the images from the class through the encoder model, and computed, which gave us their features in the R^V feature vector space. We then mapped these features to the M-dimensional vector space (R^M) to obtain the centroid for each class' embeddings. For each class C, its centroid was defined as C_n, where n ranged from 1 to k and would be further used to classify test queries.

To classify test queries, we needed to determine their geometric position in the embedding vector space. We accomplished this by passing the query images through an encoding model, which mapped them to their position in an M-dimensional space. For each image in the query set (Q), its position in the embedding space was represented as Q_λ, where λ varied from 1 to j. To classify the image, we calculated the distance between each point of the query image in the M-dimensional space and the centroids of each class, $(C_1, C_2, \ldots, C_k)$. This resulted in a matrix in which the distance of the point corresponding to the j^{th} query image from the center of the k^{th} class was represented by the index jk.

We used a simple *argmin* [14] function to calculate the distance between a query image and the centroid point of a class to predict which class would the image be classified into. We then compared the predicted class (predicted value) and the actual class (ground truth) to calculate loss and backpropogate the difference in error. To further improve the performance of the prototypical networks on the character and numeral dataset, we proposed two additional modules to be added to the original network. These were the *Weight Distribution* module, which gave weights to the initial class prototypes based on their amount of representativeness of the class; and the *Distance Scaling* module, which was able to regulate inter- and intra-class distances between the prototypes in the embedding space. While concepts similar to the Weight Distribution module have been studied in the context of Prototypical Networks before [41], we increment this approach by using an attention-based order to get representative prototypes of different classes. We describe our proposed approach in more detail in Section III-D.

D. WEIGHT DISTRIBUTION IN PROTOTYPICAL NETWORKS

One of the modules that was incorporated in the original implementation of the Prototypical Networks was the Weight Distribution module. In the original implementation of Prototypical Networks, all samples from the same class were treated equally, and were given the same weightage during the classification process. However, different characters may carry a different amount of the representativeness of the particular class that they belong to.

Taking this into consideration, we put forward an attention-based strategy to get *weighted* prototypes of different classes. This weighted prototype is obtained based on the image features of a provided sample set of images. The module consists mainly of a 3-layered network which projects the input vector to the weight space.

The following equation gives a succinct mathematical representation of the formation of the weighted instance prototypes

$$W_n = E(\kappa(X_1(J_{n1}, \ldots, J_{nk})) \tag{3}$$

Here, W_n is the weight vector for the n^{th} class in the sample set. This input provided to this module is the collective visual feature that groups instances of the same class one after the other. The module outputs a single *weight instance vector* that is perceptive of the representativeness to the class.

$$X_n = \sum_{x=1}^{x} W_{nx} * encoder(J_{nx}) \tag{4}$$

X in the above equation is the operator responsible for connecting the instance features from the same class. ξ represents the encoder that was defined previously.

E. DISTANCE SCALING

The second contribution is one that appears as another module, and aims at improving the performance of the existing vanilla Prototypical Network along with the Distance

Scaling module. To complement the self-supervised encoder and the Weight Distribution module, we propose a strategy to create a further disparity between clustered classes in the embedding space. This is achieved by maximizing the difference between classes while minimizing the distance between instances of the same class. We scale the distances with different coefficients, which are a direct product of the class-wise cross-entropy. The objective function chosen to define the distance between two instances within the same class can be written as

$$L_{\upsilon} = \sum_{i=1}^{n} log \left(\sum_{j=1}^{c} exp(-F_{\theta}(J(x_i), c_j)) \right) \tag{5}$$

Here, L_{υ} is the overall loss that is achieved after training the enhanced prototypical network pipeline after one episode, and x_i is an instance belonging to the i^{th} class. F_{Θ} is the Euclidean function that is measured as a standard metric to compute the distance in the objective function. In a similar way, the objective function that defines the inter-class distance can be formulated as

$$L_{\psi} = -F_{\theta}(J(x_i), c^{(i+1)}) \tag{6}$$

We scale both L_{υ} and L_{ψ} with different coefficients to respectively maximize and minimize them. We can hypothesize the coefficients α and β and the final equation can be formulated as

$$L = \alpha(L_{\upsilon}) + \beta(L_{\psi}) \tag{7}$$

While the above is the theoretical hypothesis of distance scaling, we consider a three-layer network to integrate this module with the larger pipeline. The feature maps generated from the instances of the *query set* are appended with the instances from the *support set*, which are finally provided to the network as the input vector. The scaling coefficient for the i^{th} class can be calculated as

$$\varepsilon = \sum_{k=1}^{K} \left(p_{\kappa}(A_{x_1,x_2}(F_{\theta}(x_{nk}), F_{\theta}(x_i))) \right) \tag{8}$$

where p is the input vector and $A_{x_1,x_2}(\circ, \circ)$ defines the append operation.

F. POST-PROCESSING

Complementing the distance scaling module, in order to achieve better accuracy on all characters of the dataset, we also had to post-process our results. This was necessary in order to account for the diacritics that the characters in the Urdu language contain. We explain this problem and our process in more detail in this section.

Each character in the Urdu language is comprised of two ligatures: the primary and secondary ligatures. In some cases, the secondary ligature may not exist. Different Urdu characters join with each other and form different ligatures based on joiner rules [29]. These ligatures often take the form of diacritics like dots or dashes that are appended somewhere with the character. We use a connected component labelling process similar to [46] in order to separate secondary ligatures from primary. To further segment the actual images, we use a simple thresholding segmentation process to look at each ligature's contour boundary. We calculate the height of each ligature and store it in an independently accessible list. Based on experiments, we classify any ligature with a height less than 30% of the largest height of all ligatures to be a secondary ligature.

These secondary ligatures are stored separately in a list and are marked during the entire training process. When the training procedure invokes the distance scaling module, we allot specific weights to different ligatures based on the joiner rules [29]. These weights allow the model to separate similar characters with only a discriminating diacritic, which helps in improving the per-class classification accuracy and the subsequent overall accuracy.

IV. EVALUATION PROTOCOL AND RESULTS

To assess the accuracy and validate that our method is correctly classifying the characters and numerals and giving a high accuracy at the same time, we applied a number of qualitative and quantitative methods to give a comprehensive overview of how our network is performing. As described above, we used two different datasets to test out our proposed approach.

A. EVALUATION PROTOCOL

This subsection deals with the method we used to estimate the accuracy of our proposed model and other models used for the comparison. A support set and query set are selected from each class. The feature vector of the images in the support set and the query set in the embedding space can be defined as the support points and query points. We classify the whole query set of every class (query points named N_Q) for all support points $N_{S=k}$ in the range $k \in [1, \ldots 5]$. The number of query points for every instance of k could be determined as $N_Q = 20 - N_S$, as we randomly sample 20 images from each class. The accuracies for all of these are aggregated, and the k-shot classification accuracy for the model can be determined as a function of k. We evaluate our models for 5-shot and 1-shot test classification.

B. DATASETS

In order to assess the relevance of our proposed approach, we tested our results on two comprehensive datasets with individually separated Urdu characters. These two datasets are the Urdu Handwritten Text Dataset (UHaT) [5], and the one used by Husnain et al. in [18] (where they used a CNN to classify individual Urdu characters). They were chosen as they correspond to the most relevant and the most recent datasets related to our task. Some recent works [2], [28] have been proposed in the literature and allow us to propose a fair comparison of our results.

The UHaT dataset [5] contains samples from more than 900 individuals who wrote characters and digits of the

Urdu language. The training set for each character contains 700 images on average. The number of images in each of the set of characters is uneven. For example: the class *alif* contains 697 images and *ayn* contains 811 images. Similarly, the training set for digits has 140 images on average. Since we are conducting the research in a few-shot environment, we randomly sampled 40 images from each class and used it as a training set; and 10 images as testing. Each of the testing images are 28×28 in size.

The other dataset we used for the experiment was collated by Husnain et al. [18]. In their paper, the authors classified Urdu characters and digits using convolutional neural networks. The dataset was built by collecting images from 500 different Urdu speakers belonging to a variety of social groups. Each of the authors was directed to write the character and the numeral in their own writing. The dataset also recorded the age, gender, hand preference while writing, physical impairments (if any) and the profession of each of the writers, but that data was not used in this experiment. As with the UHaT dataset, we randomly sampled 40 images from each class for training and 10 images from the query set each class for testing.

For both datasets, we have intentionally limited the number of samples in the training phase as this paper intends to demonstrate that it is possible to make a Urdu Handwritten Recognition system using few samples. This is done with the idea to mimic the human behavior, which is able to generalize knowledge based on few learning samples. Some examples of those datasets are provided in figure 5.

C. IMPLEMENTATION DETAILS

We conducted a large number of few-shot learning experiments on both the datasets using many random sets in order to assess the generality of the proposed approach. We used the *Adam* optimizer with an initial learning rate of 1×10^{-3}. All of our models were implemented in Tensorflow and were trained on Google Collaboratory, on a single NVIDIA K80 GPU. The training time for the Prototypical Network on each of the datasets was around a day. We trained our models for both datasets in the same way to avoid the possibility of ambiguity in the final results.

We trained our model separately on all 40 classes of Urdu characters and the 10 numeric classes and tested each trained model with 5 classes at a time (5-way classification). The reason for this was that most pre-existing literature [41], [45] demonstrated their results on the Omnilogot dataset on the basis of a 5-way classification.

D. RESULTS

This section provides a comprehensive overview on the results that we obtained from our experiments. It is divided into two major subsections, the *Qualitative Results* and the *Quantitative Results*. The Qualitative results deal with the functioning of our method and aim at illustrating the way classes are organized. This qualitative evaluation is also compared to other existing approaches, and it shows how well it is able to cluster different segments of handwritten Urdu characters with varying amounts of similarity, across common prototypes in the embedding space. In a complementary way, the Quantitative results give us an empirical overview into the accuracy of our method over the two datasets that we tested it on.

As this study tends to prove that a few-shot learning approach is able to learn the Urdu handwritten features, we trained the Prototypical Network in 1-shot and 5-shot scenarios, taking the nearest and the 5 nearest points in the embedding space. We also found that it was sufficiently advantageous to pair the training-shot with the test-shot, as well as use a higher number of classes (a higher 'way') per training episode.

1) QUALITATIVE RESULTS

As a first part of our analysis, we propose a qualitative evaluation of our proposed model. The idea behind this evaluation is to firstly illustrate that the proposed results appear relevant, and to have a visual comparison of our proposed method towards existing ones.

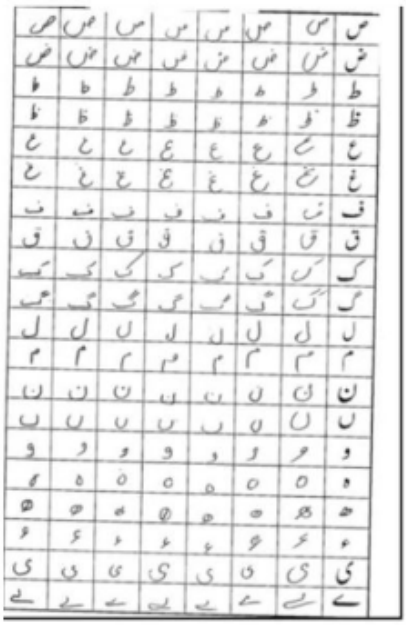
a: PCA GRAPH REPRESENTATION

To qualitatively analyze the results obtained from our proposed approach, we first developed a Principal Component Analysis (PCA) visualization of the embeddings that were learned by the prototypical networks based on the work developed in [27]. We applied PCA to reduce the embedding space to a two-dimensional slice while the network was being trained on characters. We visualized samples of characters in order to gain a better and more comprehensive insight into the learning process of the prototypical networks. Figure 6 is presenting this process on two different dataset. The left part (Figure 6a) is displaying this result on the Omnilogot dataset ([13]) while the right part (Fig. 6b) corresponds to the visualization of the embedding space obtained on the Urdu characters. Even though the visualized characters are minor variations of each other, the PCA diagram shows that the network is able to cluster hand-drawn characters closely around the prototypes, while separating the different classes.

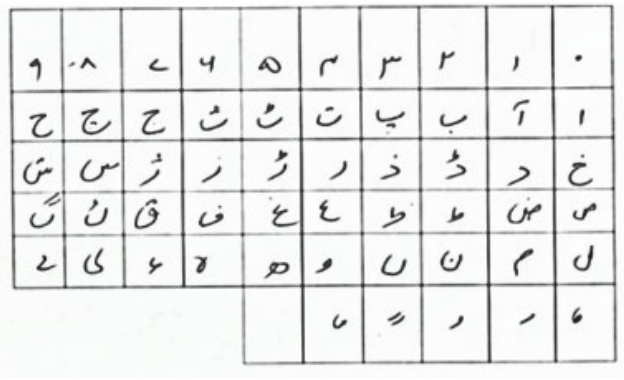
b: t-SNE COMPARISON

In order to go deeper in the visualization of our proposed approach, we propose a fair comparison with the results proposed in [19]. To do so, we used a t-distributed stochastic neighbor embedding (t-SNE) [25] to visualize our embedding space. t-SNE is a widely used non-linear dimensionality reduction technique for visualizing high-dimensional data with clear and perfect separation. Figure 7 proposed to illustrate this comparison between the results obtained in [19] and our embedding space created using only 5 samples per class.

In the previous work, the authors applied the t-SNE to visualize Urdu handwritten characters (see figure 7a). One can observe that even with a very low number of elements



(a) A sample of images in the dataset used by [19]



(b) A sample of images used in the UHaT dataset [5]

FIGURE 5. **Samples from the same set of character written by different people. The variance in each individual character is evident from the figure.**

per class (5 in this work), our proposed approach of few short learning technique (see figure 7b) is able to propose a well separated representation space. We can however see that some mistakes remains (highlighted in red circles). However, as the classification process is based on a k-nearest neighbor, those mistakes should not have a great impact on the final results.

2) QUANTITATIVE RESULTS

Following this qualitative analysis, we propose formally assessing the relevance of our proposed approach in this section. To this end, we tested it on two major datasets (Urdu Handwritten Text (UHaT) dataset [5] and the one from [18]) and we propose hereafter a comparative study against other baselines. This comparison was firstly conducted on other few-shot learning techniques, including Matching Networks [45], Neural Statistician [11] and Model Agnostic Meta-Learning (MAML) [12]. We also propose to study the genericity of the embedding models by training on one dataset and testing on the other one. Finally, we also propose a comparison with classical (*i.e.* non-few-shot learning) techniques which were published recently.

a: COMPARISON WITH FEW-SHOT LEARNING APPROACHES

The classification accuracy of our models was computed by averaging the results obtained over 200 randomly generated episodes from the testing set for both test cases (5-way-5-shot and 5-way-one-shot). We used the traditional measure of computing classification accuracy by comparing the values of the predicted class given by the network to the actual class in the dataset. We can mathematically depict the classification accuracy as:

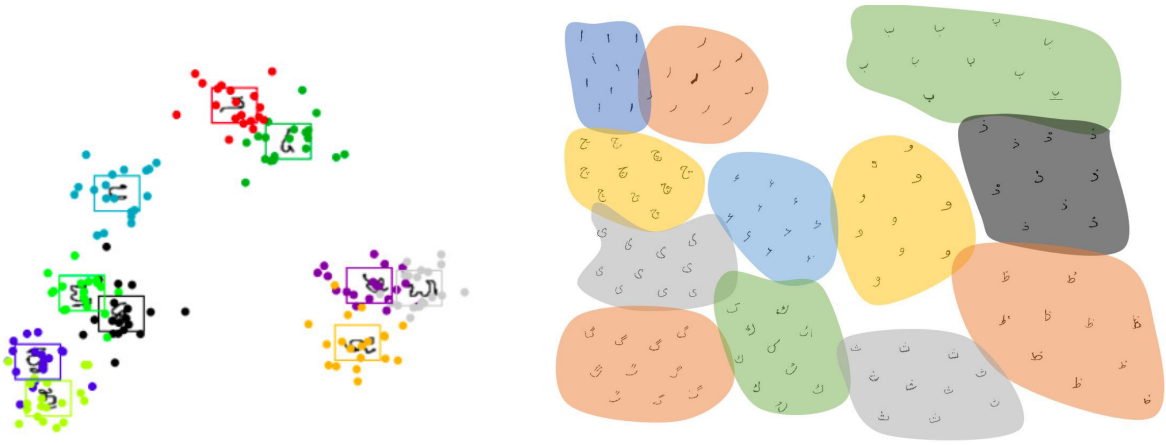
$$\Psi(y, \hat{y}) = \frac{\sum_{i=1}^{n} argmax(\hat{y}_i, 1) \sim y_i}{n} \tag{9}$$

where $\Psi(y, \hat{y})$ is the classification score for the true value y and its predicted output $\hat{y}$. n is the total number of datapoints in a randomly generated episode. We compute Ψ for 200 episodes and average the overall score to get the final accuracy for the model.

Table 1 shows a summarized view of the results on training our network on Urdu characters, while table 2 deals with Urdu Numerals. In a global overview, we can observe that our proposed method is competitive and outperforms state-of-the-art on the two datasets. In addition to computing the classwise accuracy for characters and numerals, we also compared our model against other techniques in terms of another quantifiable measure, inference time. We found that our few-shot learning approach can operate at a competitive inference time compared to other similar techniques.

Table 2 shows the classification accuracy for our method in comparison with other few-shot learning based methods, when trained on Urdu numerals separately. The classification accuracy, as with the characters, was generated by aggregating the results obtained by 200 randomly generated episodes for both the test cases, 5-way-5-shot and 5-way-1-shot. One can observe that our method is able to outperform all the existing approaches in both 1 and 5-sot learning, ensuring that the proposed approach is relevant.

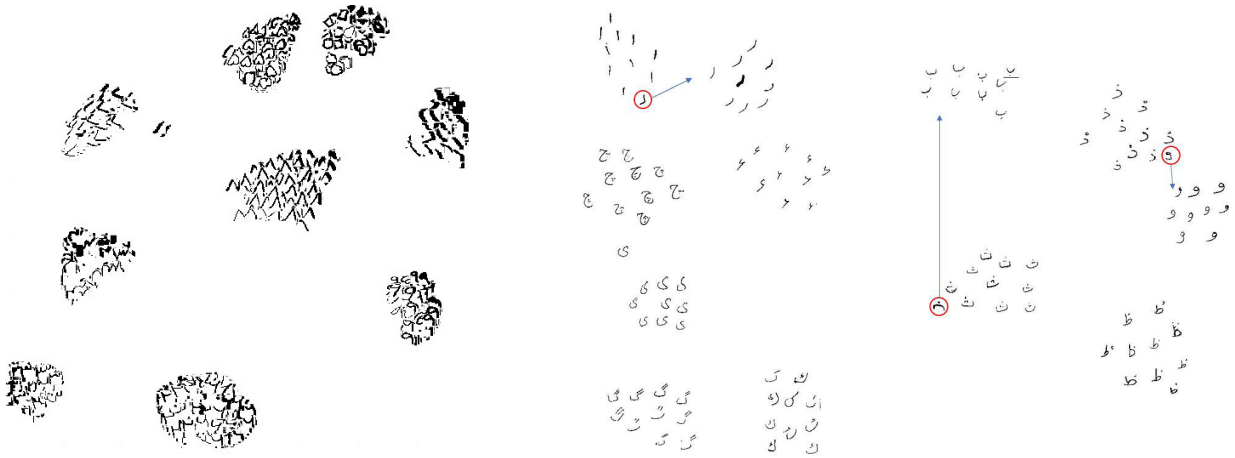
We also propose a deep analysis of our performances on how our proposed network can accurately predict a given individual character or numeral. Table 3 and Table 4



(a) PCA of embedding space taken from [13]

(b) PCA of our embedding space

FIGURE 6. A two-dimensional projection of the embedding space during training, as obtained by Principle Component Analysis. The clustering of similar characters, used for classification of unknown query images, is apparent in the plot.



(a) t-SNE of the embedding space from [19]

(b) t-SNE of our embedding space

FIGURE 7. A two-dimensional projection comparison of the embedding space unsing t-SNE during training. We can observe that few mistakes were made by our system while only using 5 samples per class for setting up the embedding space.

TABLE 1. Classification accuracy of few-shot learning methods on handwritten Urdu *characters*.

Network/Dataset	UHaT [5]		Dataset used in [18]	
Method Applied	5-Way-5-Shot	5-Way-1-Shot	5-Way-5-Shot	5-Way-1-Shot
Matching Networks [45]	91.92 ±0.5%	87.81 ±0.4%	89.22 ±0.8%	80.89 ±0.04%
Neural Statistician [11]	93.19 ±1.3%	86.31 ±0.8%	89.71 ±0.9%	81.12 ±0.3%
MAML [12]	93.08 ±0.7%	88.19 ±0.1%	91.05 ±0.4%	82.60 ±0.2%
Prototypical Network (Ours)	**94.27** ±0.2%	**88.31** ±0.3%	**91.09** ±0.6%	**82.61** ±0.1%

contain the classification accuracy for individual characters and numerals. This gives us a better insight into how well our method performs on an individual test case, rather than collectively classifying characters. An important feature of the proposed system is that in all classes, results are improved when using a 5-shot classification compared to the 1-shot classification process. This is an important feature of our model because, as displayed in figure 7b, one can see that the proposed model miss-classify some character (highlighted in red circles). Even if in a 1-shot classification process the nearest neighbor used to associate a class to a query can lead to errors and reduce the total accuracy, the 5-shot classification always proposes better results. This highlights the ability of our system to generate a well-designed representation space (*i.e.* group together similar character in the embedding space) even with very few information. While it is implicitly

TABLE 2. Classification accuracy of few-shot learning methods on handwritten Urdu *numerals*.

Network/Dataset	UHaT [5]		Dataset used in [18]	
Method Applied	5-Way-5-Shot	5-Way-1-Shot	5-Way-5-Shot	5-Way-1-Shot
Matching Networks [45]	87.77 ±0.8%	80.90 ±0.4%	89.11 ±0.3%	85.29 ±0.6%
Neural Statistician [11]	85.31 ±0.06%	80.79 ±0.8%	90.19 ±0.9%	85.21 ±1.1%
MAML [12]	85.70 ±0.9%	77.62 ±0.3%	90.70 ±0.6%	83.91 ±0.4%
Prototypical Network (Ours)	**88.57** ±0.3%	**82.41** ±0.1%	**92.89** ±0.7%	**86.41** ±0.2%

TABLE 3. Classification accuracy of few-shot learning methods on individual handwritten Urdu *characters*.

Character	5-way-5-shot (%)	5-way-1-shot (%)
alif	81.3	76.4
alif mad aa	94.3	91.2
ayn	88.8	84.5
baa	93.4	85.9
bari yaa	97.5	88.3
cheey	99.2	89.3
choti yaa	95.2	89.6
daal	96.9	91.6
dhaal	96.0	90.2
faa	92.2	82.8
gaaf	95.7	83.6
ghain	95.2	89.5
haa1	98.1	92.7
haa2	96.8	90.9
haa3	98.3	92.4
hamza	99.1	91.2
jeem	87.8	83.2
kaaf	94.2	82.3
khaa	97.9	88.6
laam	96.1	90.5
meem	90.3	85.1
noon	96.4	90.8
noonghunna	93.1	83.9
paa	88.2	79.7
qaaf	91.9	86.6
raa	83.1	77.4
rhaa	98.0	87.9
seen	90.4	84.6
seey	94.9	86.3
sheen	96.5	88.9
swaad	96.8	89.4
taa	92.2	79.5
ttaa	97.1	87.6
twa	97.3	91.4
waw	93.9	88.2
zaaa	97.6	89.9
zaal	96.2	84.3
zhaa	93.3	81.5
zwaa	95.5	88.9
zwaad	98.2	89.2

understood that the model would obviously perform better when given more data for a single class, looking at the performance of our model in multiple settings allows us to get a better understanding of its overall performance. The mean improvement when using the 5 nearest neighbor instead of the nearest neighbor is of 6,77% which clearly assess the relevance of this process.

TABLE 4. Classification accuracy of few-shot learning methods on individual handwritten Urdu *numerals*.

Numeral	5-way-5-shot (%)	5-way-1-shot (%)
0	82.2	79.8
1	81.3	77.2
2	85.3	81.2
3	78.8	73.5
4	86.4	80.9
5	87.5	82.4
6	79.2	72.3
7	86.2	83.0
8	87.9	86.6
9	86.0	84.2

Taking a step further, to even better understand the specific boundaries that our network forms when classifying characters, we decided that a Confusion Matrix would be the best way to observe the amount of similarity our network observes when comparing different characters and numerals. Thus, we created a confusion matrix from the numerals in Table 4. Figure 8 shows the confusion matrix, from which we observe that the boundaries between the correctly predicted number and a randomly selected number are quite distinct. While we did create a similar confusion matrix for all of the associated per-class accuracy of characters as depicted in Table 3, we are not including it in the current text in the interest of space. Please refer to our supplementary material to have a look at the character confusion matrix.

b: ABALATION STUDY

In order to highlight and compare the performance of our two strategies, we decided to conduct experiments and perform abalation studies, the results of which can be found in Table 5. We also conducted experiments without adding data augmentation in order to test the robustness of our approaches. We notice that our accuracy drops in both cases, signalling that data augmentation is key to our network, especially since it makes the input in a few-shot setting less sparse.

c: GENERICITY EVALUATION OF OUR APPROACH

The next part of our evaluation tends to highlight that our approach can embed some generic knowledge from one

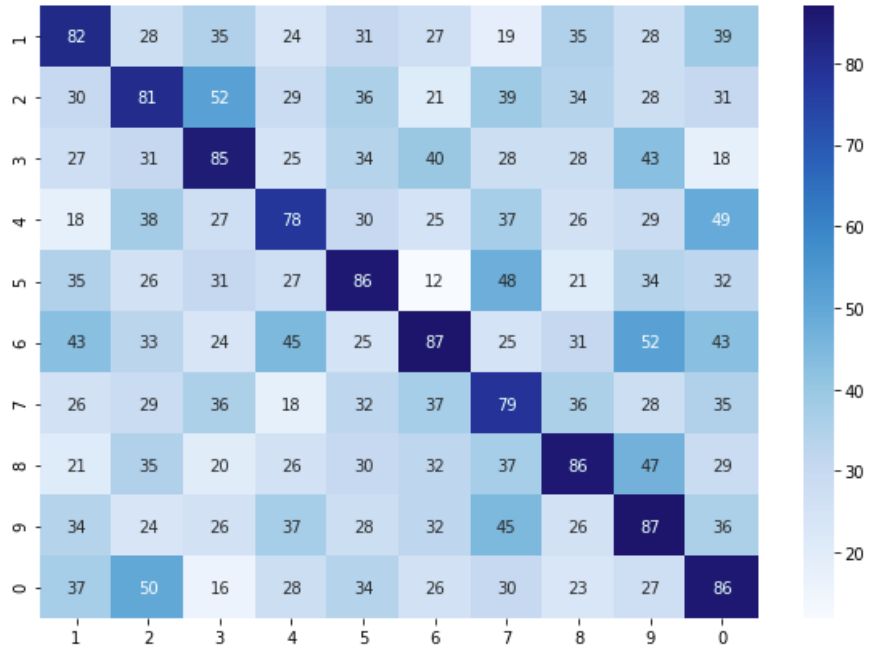


FIGURE 8. A Confusion Matrix on the 10 Urdu numerals showing the per-class accuracy and error for each of the sampled characters.

TABLE 5. Abalation study on the performance of our enhanced prototypical network. Results on 5-way-5-shot predictions of characters (A) and numerals (B). Results presented as (A/B).

Augmentation/Accuracy	With Data Augmentation	Without Augmentation
Without Weight Distribution	88% / 80%	83% / 72%
Without Distance Scaling	87% / 80%	84% / 69%
Without Both	79% / 70%	66% / 59%

TABLE 6. Genericity evalution of our few-shot learning methods on individual handwritten Numeral Urdu. The values corresponds to the classification accuracy when learning on one dataset and testing on a second one. The "Delta" values correspond to the difference with the results presented in Table 4.

Numeral	5-way-5-shot (% accuracy)	Delta	5-way-1-shot (% accuracy)	Delta
0	83.6	1.4	78.8	1.0
1	79.9	1.4	75.9	1.3
2	82.4	2.9	80.3	0.9
3	80.5	1.7	72.8	0.7
4	83.9	2.5	79.3	1.6
5	85.5	2.0	81.1	1.3
6	78.1	1.1	71.6	0.7
7	84.8	2.4	81.5	1.5
8	85.3	2.6	84.6	2.0
9	84.2	1.8	83.0	1.2

dataset to propagate it to a second one. To do so, we decided to put our model to the test by training it on a dataset and providing the query set from a completely unrelated set of images. By doing so, we were able to observe whether our model was robust to changes in the structure of the images, and was not learning shortcuts such as visual artifacts unique to a particular dataset, in order to optimize performance.

With that in mind, we trained our model on the numerals from the UHaT dataset, and we used the query set of the numerals from the dataset used in [19]. Our system reached a global accuracy of 90.81% which is approximately 1.9 points lower than the accuracy obtained on our previous result, for 5-way-5-shot learning. However, we can observe that a large part of the results remain relevant, in both 5-way-5-shot as well as 5-way-1-shot, and unchanged even with

TABLE 7. Classification accuracy of few-shot learning methods on individual handwritten Urdu numerals and characters compared with non-few shot learning methods.

Method Applied	% Accuracy (Characters)	% accuracy (Numerals)
CNN (pixel and geometrical based) [19]	96.04	98.3
Random Forest [10]	**97**	-
Support Vector Machines (SVM) [10]	**97**	-
Daubechies Wavelet [9]	-	92.05
Fuzzy Rule [34]	-	97.09
HMM [34]	-	**97.45**
Prototypical Networks (5-way-5-shot)	94.27 $\pm$0.2%	88.31 $\pm$0.3%

varied datasets. Moreover, in order to propose a fully comparable evaluation, table 6 presents a comparative study of results obtained on each number. We can observe that the changes in the accuracy have a degree of variance (denoted by *Delta*). *Delta* ranges from 1.1 to 2.9 for 5-way-5-shot learning, and from 0.7 to 2.0 for 5-way-1-shot. We can then conclude that the proposed embedding space is generic enough to be learned on one dataset and tested on another one, even with a very limiter number of samples in the training set.

d: COMPARISON WITH LARGE TRAINING MODELS

Finally, in order to determine how our method performs against existing solutions like [51], we compared our proposed approach with non-few-shot learning based approaches. This gave us an idea on how close we are to replicating the results obtained with very large set of training data used in conventional approaches [9], [10], [19], [34]. Table 7 shows the comparison between our method and other conventional handwritten character recognition methods for the Urdu language (from the second dataset [19]). While our network is not able to outperform these, it comes fairly close to emulating the results with a comparable accuracy for characters. For numerals, classical deep learning approaches using large training sets remains the best.

V. CONCLUSION AND FUTURE WORKS

A. DISCUSSIONS

Handwritten character recognition has been an age-old problem in the field of computer vision. However, most systems for this were tuned to languages like English, and French; or the languages that made use of the English alphabet. Other languages with non-English characters did not share the same limelight, which made it harder for progress to be observed. Recent developments have been made [31], [48], but the amount of structured and annotated data is usually a barrier in most cases. Rather than collating more data, we decided to use a method which was previously proven to accurately classify images with a minimal amount of examples. It is based on the idea that we can represent each character or digit as a class by the mean of its examples in the embedding space learned by a neural network. We found that our proposed approach is more efficient as well as more simpler than other meta-learning approaches, such as Matching Networks and Model Agnostic Meta-Learning (MAML), and is able to produce state-of-the art results on two independently collated datasets consisting of individual examples of Urdu characters and numerals.

In our experiment, we trained a Prototypical Network pipeline to classify handwritten Urdu characters and numerals with a minimal number of examples. We also employed an episodic training process, which was proved to be more effective for training [41], and proposed a pipeline for its training, consisting of a preprocessing step, an encoding network, the Prototypical Network acting as the classifier, and a post-processing step to account for diacritics.

Our method was able to give a good enough accuracy compared to OCR solutions for other languages [26] and was able to manage it with a minimal amount of examples. The qualitative and empirical results show that it can perform quite accurately on the basis of an OCR. However, Prototypical Networks are also prone to false positives and false negatives in some cases. This is mainly due to the fact that the embedding space relies on a small amount of data and the k-nearest neighbor procedure. While this could be a bad point for the end-user, we assume this could be an opportunity for reducing the black-box effect of such a system for non-expert users. We could then propose more than one answer to the user and illustrate those choices by displaying the embedding space, which could help in understanding the proposed answer.

B. FUTURE WORKS

In the field of handwritten text recognition, there is an abundance of data for commonly used languages like English and French. This makes character-level prediction relatively easy when compared to languages like Urdu, which, apart from having a scarcity in the amount of proper, usable data, also is bidirectional and context-sensitive in nature.

There have been a lot of developments in the field of few-shot learning in recent times. Wang et al. [48] put forward a Radical Aggregation Network for few-shot classification of handwritten Chinese characters, and we feel that the same concept would be able to give a comparable or better result than our proposed method. Other newly proposed methods like few-shot learning with geometric constraints [21], or using instance credibility inference to

improve the accuracy further [49] could also serve as viable improvements to our proposed solution. With unprecedented advancements in attention mechanisms for few-shot object detection and recognition, we believe some of the recent works by authors [7], [51] can also be adapted for the purposes of our problems. We welcome comparisons of our results with future works and it would be interesting to see new and innovative ways to tackle character recognition for non-Latin languages presented in a few-shot setting.

REFERENCES

[1] A. F. Agarap, "Deep learning using rectified linear units (ReLU)," 2019, *arXiv:1803.08375*.

[2] S. B. Ahmed, S. Naz, S. Swati, and M. I. Razzak, "Handwritten Urdu character recognition using one-dimensional BLSTM classifier," *Neural Comput. Appl.*, vol. 31, no. 4, pp. 1143–1151, Apr. 2019.

[3] Q. U. A. Akram and S. Hussain, "Ligature-based font size independent OCR for Noori Nastalique writing style," in *Proc. 1st Int. Workshop Arabic Script Anal. Recognit. (ASAR)*, Apr. 2017, pp. 129–133.

[4] M. A. A. Ali, "Language independent optical character recognition for hand written text," in *Proc. 8th Int. Multitopic Conf.*, 2004, pp. 79–84.

[5] H. Ali, A. Ullah, T. Iqbal, and S. Khattak, "Pioneer dataset and automatic recognition of Urdu handwritten characters using a deep autoencoder and convolutional neural network," *SN Appl. Sci.*, vol. 2, p. 152, Jan. 2020, doi: 10.1007/s42452-019-1914-1.

[6] H. Beigi, "An overview of handwriting recognition," Tech. Rep., 1997. [Online]. Available: https://www.semanticscholar.org/paper/An-Overview-of-Handwriting-Recognition-Beigi/f3f65927e1775013a1b201797c9a2f94a0059c80

[7] A. K. Bhunia, S. Khan, H. Cholakkal, R. M. Anwer, F. S. Khan, and M. Shah, "Handwriting transformers," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 1086–1094.

[8] W. Bieniecki, S. Grabowski, and W. Rozenberg, "Image preprocessing for improving OCR accuracy," in *Proc. Int. Conf. Perspective Technol. Methods MEMS Design*, May 2007, pp. 75–80.

[9] R. Borse and I. Ansari, *Offline Handwritten and Printed Urdu Digits Recognition Using Daubechies Wavelet*. New Delhi, India: ER, 2015.

[10] M. Chhajro, H. Khan, F. Khan, K. Kumar, A. Wagan, and S. Solangi, "Handwritten Urdu character recognition via images using different machine learning and deep learning techniques," *Indian J. Sci. Technol.*, vol. 13, no. 17, pp. 1746–1754, May 2020.

[11] H. Edwards and A. Storkey, "Towards a neural statistician," 2017, *arXiv:1606.02185*.

[12] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 1126–1135.

[13] S. Fort, "Gaussian prototypical networks for few-shot learning on Omniglot," 2017, *arXiv:1708.02735*.

[14] S. Gould, B. Fernando, A. Cherian, P. Anderson, R. S. Cruz, and E. Guo, "On differentiating parameterized argmin and argmax problems with application to bi-level optimization," 2016, *arXiv:1607.05447*.

[15] I. Haider and K. U. Khan, "Online recognition of single stroke handwritten Urdu characters," in *Proc. IEEE 13th Int. Multitopic Conf.*, Dec. 2009, pp. 1–6.

[16] Y. Hamdi, H. Boubaker, T. M. Hamdani, and A. M. Alimi, "ASAR 2021 competition on online Arabic character recognition: ACRC," in *Proc. Int. Conf. Document Anal. Recognit.* Cham, Switzerland: Springer, 2021, pp. 379–389.

[17] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997, doi: 10.1162/neco.1997.9.8.1735.

[18] M. Husnain, M. M. S. Missen, S. Mumtaz, M. Coustaty, M. Luqman, and J. Ogier, "Urdu handwritten text recognition: A survey," *IET Image Process.*, vol. 14, no. 11, pp. 2291–2300, Sep. 2020, doi: 10.1049/iet-ipr.2019.0401.

[19] M. Husnain, M. Missen, S. Mumtaz, M. Luqman, M. Coustaty, and J.-M. Ogier, "Visualization of high-dimensional data by pairwise fusion matrices using t-SNE," *Symmetry*, vol. 11, no. 1, p. 107, Jan. 2019, doi: 10.3390/sym11010107.

[20] Z. Jan, M. Shabir, M. Khan, A. Ali, and M. Muzammal, "Online Urdu handwriting recognition system using geometric invariant features," *Nucleus*, vol. 53, no. 2, pp. 89–98, 2016.

[21] H.-G. Jung and S.-W. Lee, "Few-shot learning with geometric constraints," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 11, pp. 4660–4672, Nov. 2020.

[22] N. H. Khan, A. Adnan, and S. Basar, "An analysis of off-line and on-line approaches in Urdu character recognition," in *Proc. 5th Int. Conf. Artif. Intell., Knowl. Eng. Data Bases (AIKED)*, Venice, Italy, 2016, pp. 29–31.

[23] G. Koch, R. Zemel, and R. Salakhutdinov, "Siamese neural networks for one-shot image recognition," in *Proc. ICML*, 2015, pp. 1–30.

[24] H. Liu and X. Ding, "Handwritten character recognition using gradient feature and quadratic classifier with multiple discrimination schemes," in *Proc. 8th Int. Conf. Document Anal. Recognit. (ICDAR)*, 2005, pp. 19–23.

[25] L. Van Der Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, no. 86, pp. 2579–2605, 2008. [Online]. Available: http://jmlr.org/papers/v9/vandermaaten08a.html

[26] 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.

[27] S. Mishra, U. Sarkar, S. Taraphder, S. Datta, D. Swain, R. Saikhom, S. Panda, and M. Laishram, "Principal component analysis," *Int. J. Livestock Res.*, Jan. 2017. [Online]. Available: http://ijlr.org/issue/multivariate-statistical-data-analysis-principal-component-analysis-pca/, doi: 10.5455/ijlr.20170415115235.

[28] A. Naseer and K. Zafar, "Meta-feature based few-shot Siamese learning for Urdu optical character recognition," *Comput. Intell.*, vol. 38, no. 5, pp. 1707–1727, Oct. 2022.

[29] H. Osman, K. Zaghw, M. Hazem, and S. Elsehely, "An efficient language-independent multi-font OCR for Arabic script," 2020, *arXiv:2009.09115*.

[30] A. Pal and D. Singh, "Handwritten English character recognition using neural network," *Int. J. Comput. Sci. Commun.*, vol. 1, no. 2, pp. 141–144, 2010.

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

[32] T. Rahman, *Language and Politics in Pakistan by Tariq Rahman*. Sang-E-Meel, 2011, doi: 10.1017/S0041977X0001421X.

[33] S. Ravi and H. Larochelle, "Optimization as a model for few-shot learning," in *Proc. ICLR*, 2017, pp. 1–11.

[34] M. I. Razzak, S. A. Hussain, and M. Sher, "Numeral recognition for Urdu script in unconstrained environment," in *Proc. Int. Conf. Emerg. Technol.*, Oct. 2009, pp. 44–47.

[35] N. Riaz, H. Arbab, A. Maqsood, K. B. Nasir, A. Ul-Hasan, and F. Shafait, "Conv-transformer architecture for unconstrained off-line Urdu handwriting recognition," *Int. J. Document Anal. Recognit.*, vol. 25, no. 4, pp. 373–384, 2022.

[36] M. W. Sagheer, C. L. He, N. Nobile, and C. Y. Suen, "Holistic Urdu handwritten word recognition using support vector machine," in *Proc. 20th Int. Conf. Pattern Recognit.*, Aug. 2010, pp. 1900–1903.

[37] A. Santoro, S. Bartunov, M. Botvinick, D. Wierstra, and T. Lillicrap, "One-shot learning with memory-augmented neural networks," 2016, *arXiv:1605.06065*.

[38] M. Shabir, N. Islam, Z. Jan, I. Khan, T. Rahman, A. Zeb, S. Ahmad, A. E. Abdelgawad, and M. Abdollahian, "Real-time Pashto handwritten character recognition using salient geometric and spectral features," *IEEE Access*, vol. 9, pp. 160238–160248, 2021.

[39] N. Shahzad, B. Paulson, and T. Hammond, "Urdu Qaeda: Recognition system for isolated Urdu characters," in *Proc. IUI Workshop Sketch Recognit.*, Sanibel Island, Florida, 2009, pp. 1–5.

[40] G. Singh, R. Mehta, N. Shah, and R. Mehta, "Handwriting change as a psychiatric symptom," Tech. Rep., 2016, doi: 10.19056/ijmdsjssmes/2016/v5i1/83579.

[41] J. Snell, K. Swersky, and R. S. Zemel, "Prototypical networks for few-shot learning," 2017, *arXiv:1703.05175*.

[42] S. N. Srihari, "Recognition of handwritten and machine-printed text for postal address interpretation," *Pattern Recogn. Lett.*, vol. 14, no. 4, pp. 291–302, 1993, doi: 10.1016/0167-8655(93)90095-U.

[43] I. Uddin, D. A. Ramli, A. Khan, J. I. Bangash, N. Fayyaz, A. Khan, and M. Kundi, "Benchmark Pashto handwritten character dataset and Pashto object character recognition (OCR) using deep neural network with rule activation function," *Complexity*, vol. 2021, pp. 1–16, Mar. 2021.

[44] M. A. J. Van Gerven and S. M. Bohte, "Artificial neural networks as models of neural information processing," *Frontiers Comput. Neurosci.*, vol. 11, no. 114, 2017.
[45] O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, and D. Wierstra, "Matching networks for one shot learning," 2016, *arXiv:1606.04080*.
[46] N. Vishwanath, S. Somasundaram, M. R. R. Ravi, and N. K. Nallaperumal, "Connected component analysis for Indian license plate infra-red and color image character segmentation," in *Proc. IEEE Int. Conf. Comput. Intell. Comput. Res.*, Dec. 2012, pp. 1–4.
[47] A. Wali and S. Hussain, "Context sensitive shape-substitution in Nastaliq writing system: Analysis and formulation," in *Innovations and Advanced Techniques in Computer and Information Sciences and Engineering*. Berlin, Germany: Springer 2007, pp. 53–58.
[48] T. Wang, Z. Xie, Z. Li, L. Jin, and X. Chen, "Radical aggregation network for few-shot offline handwritten Chinese character recognition," *Pattern Recognit. Lett.*, vol. 125, pp. 821–827, Jul. 2019.
[49] Y. Wang, C. Xu, C. Liu, L. Zhang, and Y. Fu, "Instance credibility inference for few-shot learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 12836–12845.
[50] Y. Wang, Q. Yao, J. T. Kwok, and L. M. Ni, "Generalizing from a few examples: A survey on few-shot learning," *ACM Comput. Surveys*, vol. 53, no. 3, pp. 1–34, May 2021, doi: 10.1145/3386252.
[51] C. Yang, Q. Wang, J. Du, J. Zhang, C. Wu, and J. Wang, "A transformer-based radical analysis network for Chinese character recognition," in *Proc. 25th Int. Conf. Pattern Recognit. (ICPR)*, Jan. 2021, pp. 3714–3719.
[52] D. Zhu, H. Yao, B. Jiang, and P. Yu, "Negative log likelihood ratio loss for deep neural network classification," in *Proc. Future Technol. Conf. (FTC)*, 2018, pp. 276–282.

RAJAT SAHAY is currently pursuing the bachelor's degree with the School of Computer Science and Engineering, Vellore Institute of Technology, Vellore, India. He was involved in computer vision techniques to solve problems in disparate fields. His research interests include applied machine learning, vision, and statistical techniques.

MICKAËL COUSTATY (Member, IEEE) is currently an Associate Professor with the Laboratoire Informatique, Image et Interaction, La Rochelle Université, France. His research interests include document analysis based on computer vision and natural language processing, digital trust in information exchange with the development of fraud detection techniques and document securing systems, segmentation and classification of large document flow, and pattern recognition.