Recognizing Bengali Word Images - A Zero-Shot Learning Perspective

Chanda, Sukalpa; Haitink, Daniël; Prasad, Prashant Kumar; Baas, Jochem; Pal, Umapada; Schomaker, Lambert

Published in:
2020 25th International Conference on Pattern Recognition (ICPR)

DOI:
10.1109/ICPR48806.2021.9412607

2020 25th International Conference on Pattern Recognition (ICPR)
Milan, Italy, Jan 10-15, 2021

Recognizing Bengali Word Images - A Zero-Shot Learning Perspective

Sukalpa Chanda*, Daniël Haitink‡, Prashant Kumar Prasad†, Jochem Baas‡, Umapada Pal†, Lambert Schomaker‡
* Department of Information Technology, Østfold University College, Norway
Email:-sukalpa@ieee.org / sukalpa.chanda@hiof.no
† Computer Vision and Pattern Recognition Unit, Indian Statistical Institute, India
Email:-umapada@isical.ac.in
‡ Bernoulli Institute for Mathematics, Computer Science and Artificial Intelligence, Faculty of Science and Engineering,
University of Groningen, The Netherlands
Email:-l.r.b.schomaker@rug.nl

2020 25th International Conference on Pattern Recognition (ICPR) | 978-1-7281-8808-9/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICPR48806.2021.9412607

Abstract—**Zero-Shot Learning(ZSL) techniques could classify a completely unseen class, which it has never seen before during training. Thus, making it more apt for any real-life classification problem, where it is not possible to train a system with annotated data for all possible class types. This work investigates recognition of word images written in Bengali Script in a ZSL framework. The proposed approach performs Zero-Shot word recognition by coupling deep learned features procured from various CNN architectures along with 13 basic shapes/stroke primitives commonly observed in Bengali script characters. As per the notion of ZSL framework those 13 basic shapes are termed as "Signature/Semantic Attributes". The obtained results are promising while evaluation was carried out in a Five-Fold cross-validation setup dealing with samples from 250 word classes.**

I. Introduction

Over the past few years, Deep Learning-based methodologies have been outperforming traditional hand-crafted feature based techniques for various classification tasks. Despite their impressive performance in different kind of classification problems, one cannot ignore the fact that the performance of Deep Learning-based methods largely depend on the availability of huge amount of annotated data samples per class. Moreover, Deep Learning based methods could only classify a data sample from one of the class type it has seen during training, but fails to classify a data sample from an unseen class. Hence such methods cannot be deployed in many real-life scenarios. On the other-hand, Zero-Shot learning based techniques are currently gaining popularity for their competence in classifying data samples coming from a completely unseen class. Such special ability makes Zero-Shot learning a natural choice for situations where new/unknown class data samples are much likely to occur. Zero-Shot learning initially has been used for object detection problem, though it could be very well utilized in applications like word image recognition for document image retrieval and document indexing as well as in postal automation where the system needs to read the address. This research investigates Bengali word image recognition problem in a Zero-Shot Learning framework where 13 basic shapes of Bengali script have been used as the "Signature Attributes" of each class under consideration. In this research, those word images resemble different place names written by different people and 250 such different place names were used as classes. Encouraging results were obtained in a five-fold cross-validation setup. To the best of our knowledge only one article [1] explored word recognition in a Zero-Shot Learning framework that deals with Latin word images. But this is the very first work on any Indic script word recognition in a Zero-Shot Learning framework.

II. Related Work

In the recent past, the field of ZSL has shown some fascinating development in the field of object recognition/detection [2], [3], [4], [5], [6], [7], [8], [9]. Those published researches divulge many interesting facts about ZSL. In [3] it has been mentioned that user-defined signature attributes lose its discriminativeness in classification because even those are semantically descriptive, they are not exhaustive. This article proposes an end-to-end model capable of learning latent discriminative features (LDF) jointly in visual and semantic space. The most interesting contribution of this proposed method is a cascaded zooming mechanism which learn features after automatically identifying the most discriminative region in an image and then zoom it into a larger scale for learning in a cascaded network structure. Thus the model can concentrate on learning features from a region with object as a focus [3]. An impressive hike in accuracy for object detection in comparison to other published work is due to [4], the proposed method uses a Graph Convolutional Network (GCN) and uses semantic embeddings and the categorical relationships to predict the classifiers. This approach takes semantic embeddings as input for each node (representing visual category). It predicts the visual classifier for each category after undergoing a series of graph convolutions. During training, the visual classifiers for a few categories are used for learning the GCN parameters. During the test phase, these filters are used to predict the visual classifiers of unseen categories [4]. In [5], the proposed method utilizes the structure of the space spanned by the attributes using a set of relations. Objective functions were customized and tailor made to preserve these relations in the embedding space, this helps to induce semanticity to the embedding space and it turns out beneficial for zero-shot learning. An earlier work similar to our approach proposes to combine deep learning and zero-shot learning technique [10]. In [10], a deep learning framework was used to generate features and consequently zero-shot learning was applied to classify between different animal, object and scenery images, three datasets namely- the Animals with Attributes dataset (AwA) [11], the aPascal/aYahoo objects dataset (aPY) [12], and the SUN scene attributes dataset (SUN) [13] were used in their experiments. In [14] attribute label embedding methods for zero-shot and few-shot learning systems were investigated and [15] proposed a method that relies on human gaze as an auxiliary information generator. A benchmark and systematical evaluation of zero-shot learning w.r.t. three aspects, i.e. methods, datasets and evaluation protocol was done in [16]. In [8] ZSL has been presented as a conditional visual classification problem. Apart from object recognition/detection ZSL has been also explored in the context of emotion recognition [17], temporal activity detection [18],instructional activities in [19] etc.

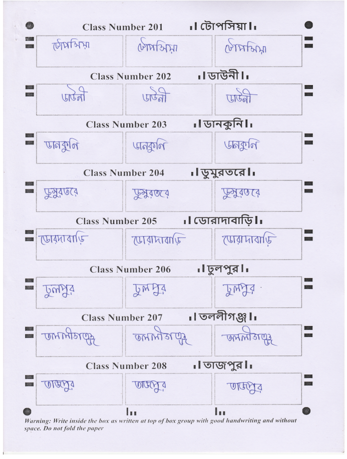
In the context of word spotting and word recognition, there have been some Deep Learning-based approaches. Sharma et al. [20] proposed a method where a pre-trained CNN is used to perform word spotting. A study by Sudholt et al. [21] proposed a novel CNN architecture for word spotting where the network is trained with the help of a Pyramidal Histogram of Characters (PHOC) representation, this work used contemporary as well as historical document images in their experiments. The proposed system can be customized as a "Query By Example"(QBE) or "Query By String"(QBS) based system.

Another deep learning-based approach for Arabic word recognition is due to [22]. In [23], the proposed method uses a deep recurrent neural network (RNN) and a statistical character language model to attain high accuracy in terms of word spotting and word indexing. Only [1] explored very briefly Latin script word recognition problem in a ZSL framework.

From the brief discussion, it is evident that though Zero-shot learning has been used extensively for animal, object and scenery image classification, it has been never used for word/text classification of any Indic script. In this paper, we exploited techniques of Zero-Shot Learning for the purpose of an "out of lexicon" Bengali script word image classification task.

III. Dataset Details and Data Collection

We considered 250 different word classes those are place names in the State of West Bengal in India. A data collection form has been prepared to obtain the handwriting sample from volunteers. Each form contains 8 classes with space (rectangular boxes) and allows the volunteer to provide 3 samples of handwriting for each class. Hence, together there are 32 such different forms to cover samples from 250 classes. Each class have at least two different handwritings. An example of such a data collection form is shown in Fig. 1.



Class Number 201 ।টোপসিয়া।

টোপসিয়া টোপসিয়া টোপসিয়া

Class Number 202 ।ডাউনী।

ডাউনী ডাউনী ডাউনী

Class Number 203 ।ডানকুনি।

ডানকুনি ডানকুনি ডানকুনি

Class Number 204 ।ডুমুরতরে।

ডুমুরতরে ডুমুরতরে ডুমুরতরে

Class Number 205 ।ডোরাদাবাড়ি।

ডোরদাবাড়ি ডোরাদাবাড়ি ডোরাদাবাড়ি

Class Number 206 ।ঢুলপুর।

ঢুলপুর ঢুলপুর ঢুলপুর

Class Number 207 ।তললীগঞ্জ।

তললীগঞ্জ তললীগঞ্জ তললীগঞ্জ

Class Number 208 ।তাজপুর।

তাজপুর তাজপুর তাজপুর

Warning: Write inside the box as written at top of box group with good handwriting and without space. Do not fold the paper

Fig. 1: Sample data collection sheet.

A. Off-line Data Augmentation

Since the number of samples/per class on this occasion was very little, we had to augment the data to bring it to a meaningful number required for training a CNN properly. We used elastic morphing technique for data augmentation purpose. In Elastic Morphing - every pixel (i,j) of the "original" image gets a random displacement vector $(\Delta x, \Delta y)$. The displacement field of the complete image is smoothed using a Gaussian convolution kernel with standard deviation σ. The field is finally rescaled to an average amplitude A. Using the displacement field and bilinear interpolation $(\hat{i} = i + \Delta x, \hat{j} = j + \Delta y)$, the new morphed image $(\hat{i}, \hat{j})$ is generated. Thus the morphing process acts on the basis of the smoothing radius σ and the average pixel displacement [24]. Total number of data samples(after the Off-line data augmentation step) used in each of the folds for training, validation and testing is depicted in Table I. Note that the training and validation data consists of 200 classes and the testing data samples constitutes data samples from 50 classes those are not present in the training and validation set.

TABLE I: Data samples used as training, validation and testing data with respect to individual folds.

Data Set For	Fold 0	Fold 1	Fold 2	Fold 3	Fold 4
Training	47360	47412	47300	47340	47370
Validation	11790	11800	11774	11780	11790
Testing	14796	14736	14868	14820	14787

IV. Motivation

The unique characteristics of Zero-Shot Learning to classify data samples from a class that has not been used for training makes it very apt to use in any real-life classification problem. For example, a word recognition system for any language would need to learn about 20k different words to be able to reach the cognitive level of any human expert in the language. It can be easily realized that in a regular supervised learning framework this will demand a huge amount of time and effort to annotate all possible words in the language. The working methodology of ZSL suits aptly for application areas like handwritten postal address recognition system as well. Place names could be considered as individual words and in an ideal situation, a ZSL-based word recognition system is expected to identify place names that it has never trained with.

V. Methodology

Despite the success of deep learning techniques in various classification tasks like object detection, speaker identification and text recognition, there are two major drawbacks of Deep Learning-based method. They are as follows: (a) It requires a considerable number of annotated training samples per class in order to achieve high classification accuracy;(b) It could only classify a test sample to any of its training classes, and is completely clueless if it encounters a class sample in the test phase that does not belong to any of "seen" class sample that it has been trained with. Real-world scenario cannot afford such pre-requisitions. For example, it is practically impossible to train a postal automation system with all possible set of address/location/place names, in such circumstances ZSL might play an important role in the future. Since deep learned features are famous for their discriminative power, this proposed method makes an attempt to combine deep learned features along with a "Zero-shot Learning" framework to counter the problem of recognizing "out of lexicon" word class images.

A. *Zero-shot learning system*

Zero-Shot Learning techniques are useful when the occurrence of some classes/objects sample to be classified during testing cannot be predicted during the training phase. Moreover, in any real-life classification problem, annotation of training samples requires human intervention in a particular domain. Zero-shot learning algorithms counter this situation by building a novel hybrid classifier with the amalgamation of a) existing classifiers and b) semantic, cross-concept mappings between class labels and visual appearance of an object class [1]. The objective of any ZSL algorithm is to obtain a mapping between the feature space and the semantic attribute space. Please refer to Fig. 2 which depicts this relationship with the help of a diagram.

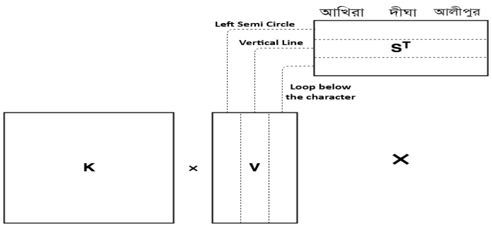


Fig. 2: The role of translation matrix V. Matrix V provides a mapping between the feature space in K and the attribute space in S.

The proposed zero-shot learning algorithm is based on *Embarrassingly Simple Zero-Shot Learning* from [10]. The system has been developed in python and will be available in a git repository if the article is accepted for publication in the future. An adapted notation scheme in comparison to [10] has been used in this article.

During the training stage, the system first creates a kernel $K \in R^{m \times m}$ from instance matrix $X \in R^{m \times d}$, here m is the number of instances and d is the dimensionality of the data. This kernel is a Gaussian kernel which depends on the hyper-parameter σ. Equation 1 shows the computation for the Gaussian kernel K. Fig. 3 depicts the computation of K if the kernel is linear in nature.

$$K(X_i, X_j) = exp\left(-\frac{\|X_i - X_j\|^2}{\sigma^2}\right) \tag{1}$$

Combining the Kernel matrix with the "attribute signature" matrix $S \in [0,1]^{z \times a}$ the algorithm computes a matrix V which performs the mapping between the feature space (represented by K), and the attribute space (represented by S), according to equation 2, z is the number of word classes, and a is the number of signature attributes (as explained in V-C), $Y \in \{0,1\}^{m \times z}$ represents the ground truth labels of each instance belonging to any of the z word classes.

$$V = (K^\top K + \gamma I)^{-1} KYS(S^\top S + \lambda I)^{-1} \tag{2}$$

In equation 2, λ and γ are hyper-parameters. Together with hyper-parameter σ, these are optimised during learning. These parameters are denoted as:

- The value of σ represents the standard deviation of the Gaussian kernel computation.
- The value of λ makes the instances on the attribute space KV more invariant. This improves the total accuracy [10].
- The value of γ balances the values of signature attribute matrix S. If the attribute values are unbalanced, the system may not perform optimally [10].

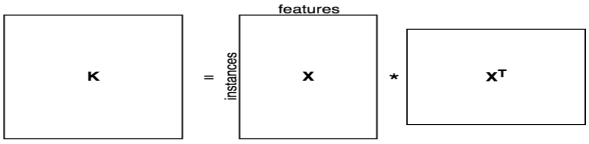


Fig. 3: Linear computation of matrix K from matrix X [10].

During training, the optimal values for hyper-parameters λ and γ and σ(only if Gaussian Kernel has been used) were determined using a grid search technique on a set of possible parameter combination values. Right at the start of the training phase, 20% of the training classes were randomly chosen as the validation set for the ZSL algorithm. Now using various combination of parameter values the model is trained using the rest of the 80% training class data samples and are tested on the validation data. The combination of hyper-parameter values that obtained the best result on the validation class samples is considered as optimal hyper-parameter values. Those optimal values for the hyper-parameters were consequently used during the testing phase. In a Zero-Shot Learning framework, correct classification means recognition of a sample image that belongs to a unseen word class. In our case, the ground truth information of "test word class" labels were already known to us by virtue of the five-fold cross validation experimental framework. Hence during the inference stage, a new set of classes (where z' denotes total number of test classes) were introduced along with their attribute signature in matrix S', and with their feature instances in matrix X'. Linearly, the kernel K' is then computed as: $K' = X'X^{\top}$. Note the relation between the training instances (in matrix X) and the testing instances (in matrix X'). Similarly, the Gaussian kernel variant of kernel K' can be computed using the optimal σ value found during training. The resulting classification is calculated per instance k in K' by plugging in those values in equation 3.

$$\underset{i}{\operatorname{argmax}}\, kVS_i'^{\top} \tag{3}$$

In equation 3, i represents the class out of z' that the instance k is classified as. The code can be found in [1].

B. Deep Learning for Feature Extraction

Deep learning techniques have been very successful in diverse image classification problems, but this success comes with the cost of training a huge number of network parameters. One of the drawbacks of AlexNet was this huge number of parameters ($\approx 62million$) that require to be trained. VGG16 evolved with a remedy to this problem. In the first and second convolutional layer of AlexNet, large kernel-sized filters of size 11 and 5 were used respectively) but in VGG16 those filters were replaced with multiple 3×3 kernel-sized filters one after another. The input to the network is a fixed size 224 × 224 RGB image. The image is propelled through a series of convolutional (conv.) layers, here filters with a 3×3 receptive fields were used. In one of the configurations, it also utilizes 1×1 convolution filters, which can be seen as a linear transformation of the input channels (followed by non-linearity) [25]. The convolution stride is fixed to 1 pixel; the spatial resolution is preserved after convolution by adjusting the spatial padding of conv. layer input. Spatial pooling is carried out by five max-pooling layers, which follow some of the conv.layers (not all the conv. layers are followed by max-pooling). Max-pooling is performed over a 2×2 pixel window, with stride 2 [25]. Any generic CNN architecture is composed of two different components:(a) the first few convolution layers of the network - that performs convolution with the help of different filters to generate features (b) the features generated in the convolution layers are propelled to the fully connected layers. Here a Multi Layer Percepton architecture is formed by stacking multiple fully connected layers. Finally, the softmax function is applied to the output of the last layer of the network to obtain the class probabilities for respective classes. Here in this case, instead of the final classification layer values, feature response of dimension 4096 from the Fully Connected layer was considered as the features from an input image. See Fig. 4 for a pictorial illustration. The network was trained using the data samples from trained and validation set in each fold as depicted in Table I, and the trained model was saved. Later using the weights of the saved model, features from images in the training and testing dataset were extracted separately and being fed to the ZSL algorithm for training and testing respectively.

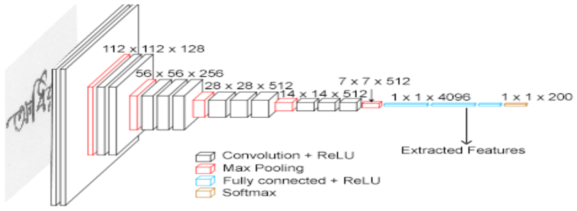


Fig. 4: Customized VGG16 Network as used in our Experiment. Original source [25].

C. Signature Attributes for Bengali Words

A "Signature Attribute" represents some unique visual/semantic characteristics of the associated class which makes a distinct mark of the difference with other classes involved in the classification process [1]. The idea is to set value for one particular "attribute signature" to 1 for all classes exhibiting that characteristics and 0 for other classes. In the context of object detection, the difference in visual/semantic characteristic between different objects seem to be very obvious like colour, shape etc. Unfortunately in images of handwritten text/words prominent visual clues are not present. Rather in case of handwritten text, different types of handwriting strokes were considered as the primitive shape structures to procure "Signature Attribute" for a particular word. A close research on Zero-Shot learning-based Latin script word recognition is due to [1]. But due to the difference in character shapes in Bengali and Latin/Roman script the "Signature Attribute" as described in [1] cannot be used here directly and needs to be adequetly customized. The "Signature Attribute" proposed in this research are as follows:(a)left semi-circle;(b)verticle line;(c)bottom semi-circle;(d)right semi-circle;(e)left top hood;(f) Diagonal line (135°), going from right to left;(g)Diagonal line (45°), going from left to right;(h)loop within a character;(i)dot below a character;(j)loop below the character;(k)horizontal line;(l)left small semi-circle;(m)right top hood. See Fig. 5, where a particular basic "Signature Attribute" shape observed in a character has been marked in red.

The "Signature Attributes" of a word class are computed by considering different shape attributes of its characters. Two variants of "class/attribute signatures" matrix were derived using different type of combinations of those 13 primitive shape attributes. Given a word class, the scores for each of those primitive shapes were computed automatically using a python script. The variants of "Signature Attribute" matrix are as follows:(a) *S-alphabet*- this is the most basic "Signature Attribute" matrix consisting of just the occurrence count of each primitive shapes, and additional 70 columns to denote

[1] https://github.com/Zero-Shot/Bengali-ZSL

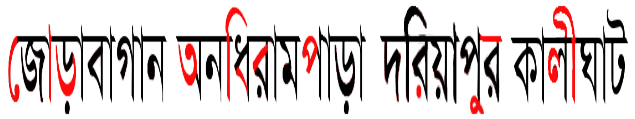


Fig. 5: The basic shape attributes marked in red (no repetetive marking of same attributes in other characters). From left to right- (a)left semi circle;(b)verticle line;(c)bottom semi-circle; (d)right semi-circle;(e)left top hood;(f)Diagonal line (135°), going from right to left;(g)Diagonal line (45°), going from left to right;(h) loop within a character;(i)dot below a character;(j)loop below the character;(k)horizontal line;(l)left small semi-circle;(m)right top hood.

presence(absence) of 70 basic alphabets of Bengali script with their occurance count. Finally, the scores are normalized by the total number of characters in the word; (b) *4S-Split-Aplhabet* - This matrix is computed by first dividing the length of the words into 4 parts and then computing scores of *S-alphabet* within each of those 4 parts. When dealing with word lengths (total number of characters in the word) that are not divisible by 4, we decide the splitting points of the word based on the lowest integer found after dividing the word length by 4 as in [1]. Hence in case of a word with a word-length that is not exactly divisible by 4, the last parts will be bigger than the first. At most, the difference will be of one character. In the case of a word with less than four characters, the first parts of the 4S matrix will remain "empty", as it will be filled with zeros. Details on each matrix type and properties are in Table II. For each word class, the "Signature Attributes" computed in *4S-Split-Alphabet* takes into account the ordering of appearance of each primitive shape/stroke in the word.

TABLE II: Number of attributes of the four versions of signature attribute matrix and their properties. S

Type of signature/attribute matrix (attribute dimension)	**Properties of the matrix**
S-Alphabet (83 is the dimension)	Scores for 13 primitive shape attributes in a word along with the count of 70 basic Bengali alphabets were computed. This signature attribute for each class is procured by counting the presence/absence of each alphabet and counting the occurrence count of each primitive stroke in the word, then dividing each value by the total number of characters in the word.
4S-Split-Alphabet(332 is the dimension of this signature attribute)	The word image is divided into 4 parts and within each part scores for 13 primitive shape attributes in a word along with the count of 70 basic Bengali alphabets were computed as done above for S-alphabet type. Hence 83×4 makes 332.

1) Methodology to Compute Signature Attribute Values: Assigning Signature Attribute of a class if calculated manually could be time consuming and error prone since we need to compute hundreds of "signature attribute" values for each class. To avoid these problem, a python script has been developed which automatically generates signature attributes per class. To generate a "signature attribute" for a particular word class this script uses a "feature alphabet" file. Each row in this file contains information about presence/absence of a particular signature attribute for a bengali character. In each row of this "feature alphabet" file, the first column represents the character itself followed by a 1(presence) or 0(absence) of a particular "signature attribute" in that character in the consecutive 13 columns of that row. Hence, considering 70 basic Bengali characters and 13 primitive shapes as "signature attribute", our "feature alphabet" file consists of 70 rows and 13 columns, where depending on the presence/absence of a particular signature attribute, the column consists value of 1/0.The python script reads through a file containing all the class labels (for each fold) in text format. Every row in the input file is occupied by a single word class label. For each word class label, the script separates the characters of the said word class. Per character, the script obtains the character's each "signature attribute" presence(absence) value from the "feature alphabet" file. A presence will add the count of occurance for that particular "signature attribute" by 1. This will be done for all characters present in the word class. When all the characters of the class are processed, the cumulative values for every "signature attribute" are normalized by dividing it with the total number of characters present in that word class. All these features are then written to a row in an output feature file. The output file contains the "signature attribute" features of that word class in one row. For 4s-Split -Alhphabet "signature attribute" generation, the word image is divided into 4 parts and above mentioned operations are performed each time within one of those 4 parts. Executable python script to compute "signature attribute" for a list of given Bengali words can be found in[2].

VI. Experimental Results, Discussion & Analysis

Experiments were conducted under several different experimental setup, which involves mainly (a) different versions of the signature attribute matrix S; (b) using different CNN architectures to extract features to be used in ZSL. Training were conducted for 100 epochs and the best model file from those CNN architectures with respect to lowest validation loss has been used to extract features.

A. Accuracy with Respect to Signature Attribute Type

Efficacy of the proposed system with respect to the Signature Attribute matrix type has been investigated. As expected, the basic (S) Signature attribute matrix performed much worse than its sophisticated counterpart (4S-Split-Alphabet). Note that in Table III, accuracy is reported with respect to each fold considering the max. out of two models those were created using two different optimization techniques for training the VGG16 network.

[2] https://github.com/Zero-Shot/Bengali-ZSL

TABLE III: Performance with Respect to Different Signature Attribute

Sig. Attri.	Fold 0	Fold 1	Fold 2	Fold 3	Fold 4
S-Alph.	23.88%	32.35%	33.15%	29.66%	19.88%
4S-Sp.-Alph.	49.89%	39.06%	48.98%	49.06%	50.53%

B. Effect of Optimization Algorithm for Feature Extraction

As mentioned earlier, deep-learned features from VGG16 network was fed to the proposed ZSL algorithm to learn the mapping between feature space and the “signature attribute” space. To train the network, two different types of optimization algorithm was explored namely (a)stochastic gradient descent;(b)Adam: A Method for Stochastic Optimization. Results with respect to those two optmization algorithms are depicted in Table IV.

TABLE IV: Performance with Respect to Different Optimization Algorithm used in Training the VGG16 Network.

Opti. Algo.	Fold 0	Fold 1	Fold 2	Fold 3	Fold 4
Adam	49.18%	39.06%	48.98%	49.06%	50.53%
SGD	49.89%	38.69%	48.24%	47.09%	48.54%

Network training parameters with respect to SGD and ADAM optimization algorithm used are as follows: For SGD - $learning-rate = 0.00001, decay = 1e-6, momentum = 0.9.$, and for ADAM - $learning-rate = 0.00001, beta_1 = 0.9, beta_2 = 0.999, epsilon = 1e-8$.

C. Performance Analysis With Different CNN Architectures for Feature Extraction

To analyze efficacy of features from other standard CNN architectures like ResNet152, XceptionNet and GoogleNet, different experiments were conducted. All network architectures were imported as modules from Keras library and takes an input of dimension 244 width and 150 height. Those networks were trained from scratch using the same data(maintaining the respective fold) as we did for VGG16 along with “Adam Optimizer” with default values. Once the training is complete, best model files with respect to minimum validation loss were used to extract features from the “avg-pool” layer of those network models. Those extracted features(of dimension 2048) were fed to the proposed ZSL algorithm to learn the mapping between feature space and the “signature attribute” space. Results with respect to features generated from different CNN architectures are depicted in Table V.

TABLE V: Performance with Respect to Different CNN architecture as the Feature Extractor.

Archi.	Fold 0	Fold 1	Fold 2	Fold 3	Fold 4
GoogleNet	35.09%	41.32%	30.28%	28.64%	39.66%
ResNet152	29.26%	28.52%	35.88%	26.07%	27.36%
XceptionNet	44.76%	35.45%	41.43%	38.21%	44.57%

D. Attentive Region Embedding Network(AREN)- Based Zero-Shot Word Recognition

The proposed method is based on “EZSL” [10] a shallow model which takes pre-trained features as input, while AREN is an end-to-end model which takes raw image as an input. Earlier ZSL algorithms used to leverage the global features from which such algorithms used to construct the semantic space embedding. But another approach has currently evolved in the ZSL spectrum. This newly evolved approach advocates to study the discrimination power implied in local image regions (parts), and hence assist the semantic transfer between seen/unseen classes. AREN [9] comes under this catagory. AREN follows an end-to-end trainable approach and consists of two network branches, i.e., the attentive region embedding (ARE) stream, and the attentive compressed second-order embedding (ACSE) stream. AREN is capable of discovering multiple part regions under the guidance of the attention and the compatibility loss. Moreover, a novel adaptive thresholding mechanism is proposed for suppressing redundant (such as background) attention regions. Since this method has outperformed many other methods on various Zero-Shot benchmark datasets, we were curious to check the effectiveness of AREN in terms Zero-Shot handwritten word recognition. Keeping this objective in mind, the code of AREN (available through a github repository of the authors) has been amended so that it can deal with Zero-Shot word recognition problem by considering our set of “Signature Attributes” those are pertinent for Bengali word recognition. In AREN the feature generating backbone is basically a ResNet network architecture. A striking phenomena to notice here is that the results are same or a bit inferior while we used a vanilla ResNet to extract features, hence we can say that the Attentive region highlighting technique of AREN which worked very well for regular Zero-Shot recognition of object/animal,failed in case of Zero-Shot word recognition problem. The possible reason for this could be the lack of decisive visual features like colour and texture based characteristics in the handwritten text feature space, which contributes significantly in case of object/animal recognition. Results on individual folds using AREN method is depicted in Table VI.

TABLE VI: Performance with AREN method

Fold Number	Accuracy
0	26.41%
1	27.24%
2	31.61%
3	25.11%
4	30.31%

E. Comparison with previous research

To the best of our knowledge, no work till date exists on Indic script word recognition using a Zero-Shot Learning framework. Hence a one-to-one comparison is not possible. The closest research is due to [1], but that deals with medieval Roman script. Though a direct comparison is not pertinent in this context, an effort has been made to get an idea about the performance range that current ZSL systems exhibits. Here in Table VII, (Tr) denotes training and (Te) denotes testing. The experiments from Xie.et.al. [9] shows obtained results on some standard ZSL performance benchmark datasets. All those benchmark datasets either consists of huge number of samples/per class or huge number of training classes. For example, (a) AwA2, consists of 37,322 images of animals from 50 classes; (b) aPY consists of 15,339 images from only 32 classes;(c)CUB consists of 11,788 bird images from 200 classes,(d) SUN consists of huge number of training classes. Note that all datasets those are used for benchmarking a ZSL algorithm consists of animals and other objects where one could get the opportunity to describe those animals/object in the semantic space using a wide range of visual semantic attributes. This is certainly not possible in case of word images. It is worth mentioning that a state-of-art ZSL algorithm such as [9] failed to obtain accuracy higher than 39.2% even while the number of unseen test classes were as less as 10. One must not ignore the fact that datasets (like AwA2,CUB and SUN) used for performance benchmarking of ZSL algorithms are all animal/bird/object image dataset and hence the semantic attribute space for such a dataset could be very rich. The proposed system cannot be evaluated on those benchmark datasets as the proposed signature attributes to represent the semantic attribute space of word images is completely different from signature attributes used for those benchmark datasets. In ZSL benchmark datasets (AWa2, APY etc), there are certain advantages like: (a) huge number of samples/class;(b) smaller number of test classes in comparison to number of training classes;(c) very broad semantic attribute space can be associated with classes in those datasets, which is not the case with the type of data we are dealing with. Hence, we can claim that we achieved comparable results in Bengali script Zero-Shot word recognition problem. The average accuracy for the proposed system has been calculated by considering the best accuracy in each folds from Table III.

TABLE VII: Out of lexicon performance using ZSL (From left to right, first 3 columns regular image datasets on animals, objects, results from [9]), handwritten latin script word images and bengali handwritten word images.(Accuracy: top-1 in %)

	AwA2	aPY	SUN	CUB	*Medieval Roman Script Word Recognition*	*Proposed System for Bengali Script*
Tr. Classes	40	20	645	150	166	$200/fold$
Te. Classes	10	12	72	50	50	$50/fold$
Accuracy	67.90	39.20	60.70	70.08	57 ≈	47.50

F. Error Analysis

Experiments were conducted considering 4 different CNN architectures(VGG,ResNet,GoogleNet and XceptionNet) for extracting features from word images. It was being observed that the accuracy for "fold 1" test dataset was always the lowest while using VGG and XceptionNet as the feature extractor. These intruiging phenomena could possibly due to the dfference in relationship between Signature Attributes(4S-Split-Alphabet) of train, test and validation classes in fold 1 and the rest of the folds. It has been explained with the

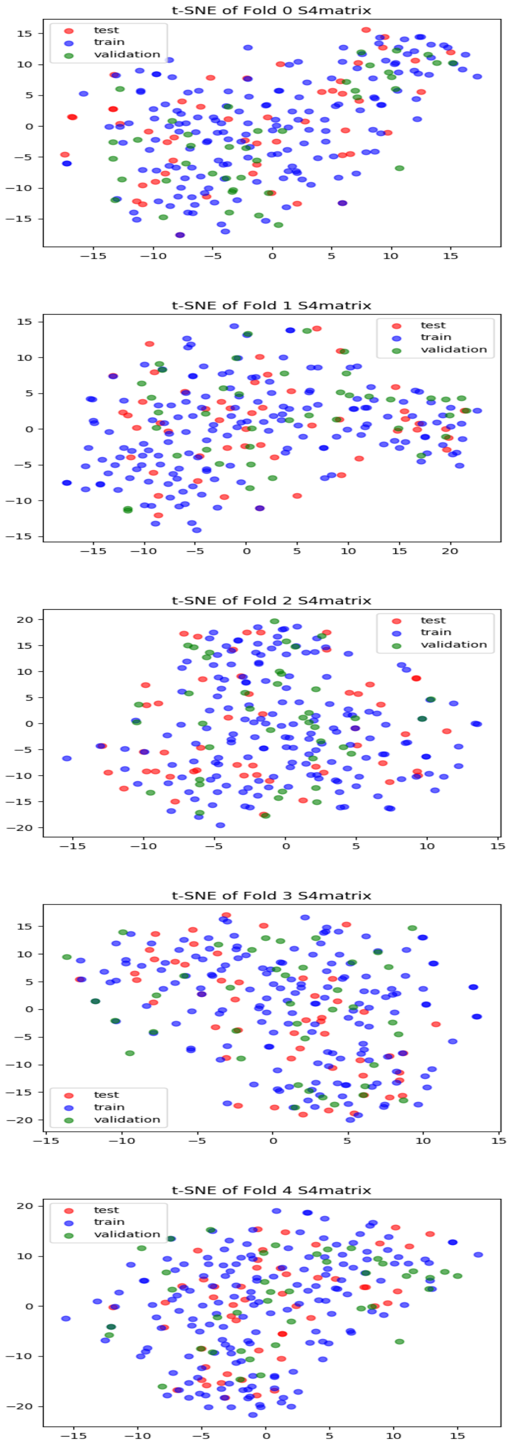


Fig. 6: From top till bottom TSNE-plot of Signature Attributes(4S-Alphabet) for training, testing and validation classes with respect to individual folds in increasing fold number.

help of Fig. 6. Here with the help of TSNE (t-Distributed Stochastic Neighbor Embedding, which is an unsupervised, non-linear technique primarily used for data exploration and visualizing high-dimensional data), Signature Attributes(4S-Split-Alphabet) for training, testing and validation classes with respect to individual folds were projected in a 2-d space. As depicted in figure 6, it can be easily noted that in case of the tsne plot for "fold 1", the signature attributes ranges from -15 to +15 in the Y-Axis and -15 to $\approx +21.5$ in the X-Axis, which is not the case with the rest of the 4 tsne plots. In the rest of the 4 folds, the data samples are in the range of -15 to $\approx +15$ in the X-Axis and -20 to +15 in the Y-Axis. One more noteworthy issue is that in the tsne plot of "fold 1", the bottom left corner region barely consists of any test and validation data samples, and is dominated by mainly training data samples. Whereas in the rest of the four diagrams - train, test and validation data samples are scattered in all regions. It can be noted that for ResNet and GoogleNet extracted features, the effect of "Signature Attributes" is not very obvious in terms of system accuracy, and no such clear trait can be observed as we have seen in case of VGG16 and XceptionNet.

VII. Conclusion & Future Works

This research investigates recognition of Bengali script word images in a ZSL framework. Though ZSL has been largely utilized for object detection/recognition purpose in the past, its usage for an Indic script word recognition has never been investigated before. Different strokes found in Bengali alphabets has been used as "signature attributes" for each class and promising results were obtained. Analysis shows that quality of the signature attributes used in a ZSL framework plays a crucial role in obtaining reasonable accuracy. Future research could be in the direction of finding more robust signature attributes and also to investigate the possibility of procuring a script independent "signature attributes" for multiple Indic scripts.

References

[1] Sukalpa Chanda, Jochem Baas, Daniel Haitink, Sébastien Hamel, Dominique Stutzmann, and Lambert Schomaker, "Zero-shot learning based approach for medieval word recognition using deep-learned features," in *16th ICFHR,USA*, 2018, pp. 345–350.

[2] Xiaolong Wang, Yufei Ye, and Abhinav Gupta, "Zero-shot recognition via semantic embeddings and knowledge graphs," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[3] Yan Li, Junge Zhang, Jianguo Zhang, and Kaiqi Huang, "Discriminative learning of latent features for zero-shot recognition," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[4] Hongguang Zhang and Piotr Koniusz, "Zero-shot kernel learning," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[5] Yashas Annadani and Soma Biswas, "Preserving semantic relations for zero-shot learning," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[6] Long Chen, Hanwang Zhang, Jun Xiao, Wei Liu, and Shih-Fu Chang, "Zero-shot visual recognition using semantics-preserving adversarial embedding networks," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[7] Li Niu, Ashok Veeraraghavan, and Ashutosh Sabharwal, "Webly supervised learning meets zero-shot learning: A hybrid approach for fine-grained classification," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[8] Kai Li, Martin Renqiang Min, and Yun Fu, "Rethinking zero-shot learning: A conditional visual classification perspective," in *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[9] Guo-Sen Xie, Li Liu, Xiaobo Jin, Fan Zhu, Zheng Zhang, Jie Qin, Yazhou Yao, and Ling Shao, "Attentive region embedding network for zero-shot learning," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[10] Bernardino Romera-Paredes and Philip Torr, "An embarrassingly simple approach to zero-shot learning," in *Proceedings of the 32nd International Conference on Machine Learning*, Francis Bach and David Blei, Eds., Lille, France, 2015, vol. 37 of *Proceedings of Machine Learning Research*, pp. 2152–2161.

[11] CH. Lampert, H. Nickisch, and S. Harmeling, "Learning to detect unseen object classes by between-class attribute transfer," in *CVPR 2009*, Piscataway, NJ, USA, June 2009, Max-Planck-Gesellschaft, pp. 951–958, IEEE Service Center.

[12] Ali Farhadi, Ian Endres, Derek Hoiem, and David Forsyth, "Describing objects by their attributes," in *Conference on Computer Vision and Pattern Recognition 2009,USA*. 2009, pp. 1778–1785, IEEE Computer Society.

[13] G. Patterson and J. Hays, "Sun attribute database: Discovering, annotating, and recognizing scene attributes," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 2751–2758.

[14] Zeynep Akata, Florent Perronnin, Zaïd Harchaoui, and Cordelia Schmid, "Label-embedding for image classification," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 7, pp. 1425–1438, 2016.

[15] Nour Karessli, Zeynep Akata, Bernt Schiele, and Andreas Bulling, "Gaze embeddings for zero-shot image classification," in *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, 2017, pp. 6412–6421.

[16] Yongqin Xian, Bernt Schiele, and Zeynep Akata, "Zero-shot learning - the good, the bad and the ugly," in *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, 2017, pp. 3077–3086.

[17] Chi Zhan, Dongyu She, Sicheng Zhao, Ming-Ming Cheng, and Jufeng Yang, "Zero-shot emotion recognition via affective structural embedding," in *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[18] Lingling Zhang, Xiaojun Chang, Jun Liu, Minnan Luo, Sen Wang, Zongyuan Ge, and Alexander Hauptmann, "Zstad: Zero-shot temporal activity detection," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 875–888.

[19] Fadime Sener and Angela Yao, "Zero-shot anticipation for instructional activities," in *The IEEE International Conference on Computer Vision (ICCV)*, 2019, pp. 862–871.

[20] Arjun Sharma and K. Pramod Sankar, "Adapting off-the-shelf cnns for word spotting & recognition," in *13th International Conference on Document Analysis and Recognition, ICDAR 2015, Nancy, France, August 23-26, 2015*, 2015, pp. 986–990.

[21] Sebastian Sudholt and Gernot A. Fink, "Phocnet: A deep convolutional neural network for word spotting in handwritten documents," in *15th International Conference on Frontiers in Handwriting Recognition, ICFHR 2016, Shenzhen, China, October 23-26, 2016*, 2016, pp. 277–282.

[22] Alex Graves and Juergen Schmidhuber, "Offline handwriting recognition with multidimensional recurrent neural networks," in *Advances in Neural Information Processing Systems 21*, D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, Eds., pp. 545–552. 2009.

[23] Théodore Bluche, Sebastien Hamel, Christopher Kermorvant, Joan Puigcerver, Dominique Stutzmann, Alejandro Héctor Toselli, and Enrique Vidal, "Preparatory KWS experiments for large-scale indexing of a vast medieval manuscript collection in the HIMANIS project," in *14th IAPR International Conference on Document Analysis and Recognition, ICDAR 2017, Kyoto, Japan, November 9-15, 2017*, 2017, pp. 311–316.

[24] Marius Bulacu, Axel Brink, Tijn van der Zant, and Lambert Schomaker, "Recognition of handwritten numerical fields in a large single-writer historical collection," in *10th ICDAR*.

[25] https://neurohive.io/en/popular networks/vgg16/, *VGG16*, 2020 (accessed Feb 3, 2020).