Indian Journal of Natural Sciences *www.tnsroindia.org.in ©IJONS*

Vol.13 / Issue 75 / December / 2022 *International Bimonthly (Print)* *ISSN: 0976 – 0997*

RESEARCH ARTICLE

A Novel Generalized Handwritten Character Recognition Model using Few-Shot Learning

Prabavathy Balasundaram[1]*, Lekshmi Kalinathan[1], Sushaanth Srinivasan[2], Sharvesh Shankar[2] and W Angeline Hilda[3]

[1]Associate Professor, Department of Computer Science and Engineering, Sri Sivasubramania Nadar College of Engineering, Chennai, Tamil Nadu, India.
[2]UG Student, Department of Computer Science and Engineering, Sri Sivasubramania Nadar College of Engineering, Chennai, Tamil Nadu, India.
[3]PG Student, Department of Computer Science and Engineering, Sri Sivasubramania Nadar College of Engineering, Chennai, Tamil Nadu, India.

Received: 02 Sep 2022 Revised: 02 Oct 2022 Accepted: 09 Nov 2022

***Address for Correspondence**
Prabavathy Balasundaram
Associate Professor,
Department of Computer Science and Engineering,
Sri Sivasubramania Nadar College of Engineering,
Chennai, Tamil Nadu, India.
Email: prabavathyb@ssn.edu.in

ABSTRACT

Handwritten character recognition is a fundamental and most challenging research area with many useful applications. Many research works have been conducted with the use of deep learning networks in developing handwritten recognition systems that are focused on building language-specific models. However, building language-specific hand-written character recognition models is a time-consuming process. Furthermore, deep learning models require a lot of labeled data and are incapable of identifying samples of unseen classes. To address the issues of building time-constrained models and building models with few samples, this paper proposes a generalized model to recognize the characters of multiple languages using few-shot learning which can train models with few samples and generalize for samples of unseen classes.

Keywords: Handwritten character recognition, Few shot learning, Siamese Network, Triplet Network, Convolutional Neural Network

INTRODUCTION

Handwritten character recognition is a field of research in Artificial Intelligence, computer vision, and pattern recognition. It is required for many diverse applications such as digitizing historical documents, processing bank cheques and huge documents in legal, healthcare, and finance. Humans can easily understand different handwriting using their intelligence and learning. The same ability is induced into machines using Deep Learning. However, it requires a large amount of labeled data which is expensive to produce and requires high-end infrastructure to train in a reasonable time. Samples of handwritten characters corresponding to alphabets of multiple languages are limited. Hence, it is necessary to use the Few-Shot Learning technique which has the capability to learn from a small number of samples. The problem of handwritten character recognition has been approached from multiple angles in the field of deep learning. Frequent works have been done only on specific languages such as Arabic, Chinese, Devanagari characters, and numbers. A few works exist on specific Indian languages and Greek [1][2][3]. Many handwritten character recognition models were built using Convolutional Neural Networks (CNNs) which uses a lot of data for training and testing and thus makes it time consuming. A large amount of existing work regarding Few-Shot Learning is related to datasets other than Handwritten Characters such as Leaf [4], Plant Disease [5], Face [6][7], Palm Vein [8] and Metal Defect Detection [9]. Few works on Handwritten Character Recognition using Few-Shot Learning supports only the alphabets in Omniglot dataset. If the same model is used for Tamil or English, it might still recognize the characters, but with low accuracy. Building language specific models [10][11][12][13][14][15] for this purpose is tiresome as it demands high computational time and heavy domain knowledge. Therefore it necessitates a more generalized model, which supports the recognition of the languages in Omniglot, Tamil, the universal language English and the digits in MNIST.
Hence the objectives of the proposed work are as follows:

- To build a generalized model to recognize handwritten characters using a Few-Shot learning approach that could identify characters from other languages including Omniglot.
- To evaluate the built model with datasets of different alphabets and analyze its results.

Literature Survey

Koch et al., [3] have first proposed a siamese neural network for one shot image recognition. The Omniglot dataset was augmented using small affine distortions and the model and an accuracy of 92% was achieved. Wang B. *et al.*, [4] have proposed a few shot learning methods based on the Siamese network for leaf classification. Feature extraction and spatial structure optimization were used. Experimental results show that with 20 training samples, classification accuracies are 95.32%, 91.31%, and 91.75% for Flavia, Swedish and Leaf snap datasets, respectively. Argueso D. *et al.* [5] have proposed a Siamese network based few shot learning with triplet loss using pretrained Inception-v4 for plant disease classification. The network is fine-tuned for plant leaf classification of the Plant Village dataset images. SVM classifier is used to classify the plant disease based on the distance between the feature vectors with an accuracy of 90%.Devi P. R. *et al.*, [6] have proposed Few-Shot learning based on the Siamese network for facial recognition. Feature vectors were extracted using CNNs and clustering was performed with the K Means algorithm. The model was able to predict well for ATT and Yale datasets. Heidari M. *et al.*, [7] have proposed face recognition using a Siamese network with contrastive loss. A pre-trained VGG16 model is used to extract the feature and Euclidean distance is calculated between the feature vectors. Experimental results show an accuracy of 95.62%. Marattukalam F. *et al.*, [8] have proposed Siamese network based few shot learning for contactless biometric systems.

The features of both the palm vein images are extracted using CNN and concatenated to obtain a feature vector. Contrastive loss with a 2-way 5-shot learning setting obtained an accuracy of 90.5%. A feature fusion model utilizes the focus-area location and high-order integration to generate the feature representation for the few-shot tasks. Kim M. S. *et al.,* [9] have proposed a few-shot learning method with a Siamese network with contrastive loss to classify the steel surface defects. Li H. *et al.,* [12] have proposed a Siamese network based one shot Chinese character recognition. CNN with multilayer feature extraction and batch normalization was used to construct the siamese network. An accuracy of 95% is achieved for five-way one-shot learning. Chakrapani G. V. A. *et al.*, [16] have proposed a solution with one shot learning for handwritten word recognition. Experiments were conducted on the George Washington dataset and the Indian City Names dataset. Data augmentation techniques were applied to avoid over fitting. An accuracy of 92.5% was achieved on five-way one-shot tasks. Mukhtar *et al*., [17] have proposed one-shot learning based on the Siamese network to recognize wheat diseases. MobileNetv3 is used to extract features that are fine-tuned with the Plant Village dataset. The last two dense layers are fine-tuned with the Computer Vision for Crop Disease (CGIAR) dataset. The extracted feature vector's absolute difference is found, and a dense layer with a sigmoid function is used to generate the similarity score. The MobileNetv3 model has achieved 98% and 96% accuracy for training and validation. The one-shot network has achieved more than 92% accuracy, 84% precision, and 85% recall.

MATERIALS AND METHODS

This section describes in detail the Few-Shot learning technique and the proposed methods to create a generalized model.

Few-Shot Learning

Traditionally, a deep learning model considers the same categories of data for both train and test datasets. For example, to build a model to classify dogs and otters, firstly, the model is trained with a massive amount of labeled data of dogs and otters, and it is tested with unseen images of the trained class. If a classification of a tiger is to be included, then the model has to be trained again with a massive amount of labeled classes of dog, otter, and tiger. Whereas in Few-Shot learning, the model is trained with limited data and also used to classify the data samples of unseen classes. This is due to the fact that the Few-Shot learning model learns to find the similarity and dissimilarity between the images given to it. A Few-Shot learning model can have different categories of data in training and test sets. For example, to build a model to classify dogs and otters, firstly, the model is trained to identify the similarities and dissimilarities between the labeled data of dogs and otters, and it is tested with images of unseen class which may or may not be of the trained class. The model built using Few-Shot learning is tested with a support set and a query set. The support set contains N classes, in which each class has K samples. Query set is nothing but a test set consisting of a set of images to be tested. If the model is queried with an image of a squirrel, a support set consisting of an image of a squirrel has to be given. Since the model is capable of identifying similar and dissimilar images, it finds the similarity scores between every query image and the images present in the support set as shown in Figure 1. Further, the model reports the class of the query image based on the highest similarity score.

Proposed Siamese Neural Network

The proposed Siamese Neural Network (SNN) shown in Figure 2. is composed of identical twin networks, which are CNNs. These networks have the same configuration with the same parameters and weights. SNN is built to learn the similarity between the handwritten characters. Hence in order to build the model, Algorithm 1 has been used to construct similar and dissimilar pairs of characters. Figure 3 shows the procedure in which similar and dissimilar pairs are constructed for a batch size of 10. Initially, ten random class numbers of characters have been generated and stored in Input_1 & Input_2. It is seen that five class numbers across Input_1 & Input_2 are to be the same. Random samples for every class are generated and stored in sample_1 & sample_2. For example, 411 from Input_1 and 728 from Input_2 represent different classes. Sample 0 from the class 411 and sample 4 from class 728 are chosen at random to generate a dissimilar pair which is represented by 0 in the targets array. Similarly, samples 8 and 11 from class 919 are considered as a similar pair which is represented by 1 in the targets array. Figures 4 and 5 show the generated similar and dissimilar image samples.

In order to train the proposed network, similar and dissimilar pairs of images x1 and x2 are fed into the twin networks. Each CNN layer utilizes filters of increasing complexity. Every layer contributes to the learning of consequent image parts throughout. The first few layers learn basic features such as horizontal lines, vertical lines, edges and corners. The middle layers detect more specific features of the handwritten character such as its curves and shapes. The last few layers learn the higher representations to recognize the full handwritten character, in different shapes and positions. Dropout layers are introduced to avoid over fitting by randomly dropping the connections between neurons. After the sequence of convolutional layers, the output is flattened into a single dimension vector and brought down into a feature vector of size 1 x 512. The feature vectors f(x1) and f(x2) are passed into the differencing layer to find the distance between them using the Euclidean Distance metric. This distance value is passed to the output layer where a sigmoid function is applied on the distance value which returns a number between 0 and 1. Based on the target and the output values, the loss will be calculated and will be used for back propagation to update the weights during training.

Proposed Triplet Neural Network:

The Triplet Neural Network (TNN) shown in Figure 6 utilizes three images as an input. These three images are a combination of an anchor image, a positive image, and a negative image. The image from the same class as the anchor image is considered to be positive and a different class is considered to be negative. For example, considering the image of a tiger to be an anchor image, any image from the tiger class is considered as a positive image. Any image other than a tiger is considered to be the negative image. Algorithm 2 has been used to construct triplets. Figure 7 shows the procedure in which the triplets are constructed for a batch size of 10. Initially, ten different random class numbers of characters have been generated and stored in Anchor & Input_2. The arrays Anchor and Input_3 are the same. Random samples for every class are generated and stored in sample_1, sample_2 and sample_3. An example of one of the triplets formed from Figure 7 consists of 411 from Anchor, 919 from Input_2 and 411 from Input_3 and are considered to be the anchor, negative and positive images respectively. Figure 8 shows the generated sample triplets. These images are fed into the respective neural network to extract the corresponding features vectors. These feature vectors of the positive and the anchor image are passed into one differencing layer to find the distance between them using the Euclidean Distance metric. Similarly the feature vectors of the anchor and the negative image are parallely passed to another differencing layer. The outputs of both the differencing layers are used to find the triplet loss. Based on the target and the output values, the loss will be calculated and will be used for back propagation to update the weights during training.

RESULTS AND DISCUSSION

This section describes in detail the datasets used for experimentation and discusses the results of the proposed methods.

Description of Omniglot Dataset

It contains 1,623 different handwritten characters from 50 different languages with 20 samples for each character.

Description Tamil Dataset

It consists of 156 characters, and each character contains approximately 300 samples written by native Tamil writers including school children, university graduates, and adults from the cities across Tamil Nadu, India. The images are in TIFF format, and the dimensions of each image are different.

MNIST dataset

The MNIST handwritten digits dataset consists of 60,000 images of handwritten single digits between 0 and 9. MNIST Fashion dataset consists of 60,000 images with 10 different classes.

Performance metric used

Accuracy - Metric for evaluating classification models. It is the fraction of correct predictions our model out of total predictions. Formally, accuracy has the following definition:

Implementation

The Omniglot dataset was used to build SNN and TNN. Among 50 languages in Omniglot, 30 languages are selected for training and 20 for testing. The train set contains 964 characters of different languages and a total of 19,280 samples. Among that, 20% of the data, i.e., 192 characters, is considered for validation. The test set contains 659 characters and a total of 13,180 samples. Pair of images or triplets were passed to SNN and TNN respectively. In both the networks, L1-distance was used to find the distances between similar and dissimilar pairs. This is fed into the dense layer with sigmoid activation and the final layer outputs the value 0 to 1, where 0 represents no similarity and 1 represents full similarity. Binary cross entropy and Triplet loss functions were used in SNN and TNN respectively. The Adam optimizer updates the weights to reduce the loss incurred.

Experiments on the proposed SNN and TNN models

The proposed SNN was evaluated by generating a random test set and the corresponding support set from the Omniglot dataset. The test set consists of classes that are not present in the train set. Figure 9 shows the test image and the support set containing 20 characters with one image per character including the test image. N such test samples are generated, and the model's accuracy is measured by the number of correct predictions made. Several variants of the proposed SNN models were built and training and testing accuracies have been measured. These models have been built with 128 and 256 image pairs, in which the model with 256 image pairs performed better compared to the other, with an improvement in accuracy of 3.2% as shown in Table 1. With the fixation of 256 image pairs, the variants of the model with 3000 and 7000 iterations have been built. The model trained for 7000 iterations performed better with an improvement in accuracy of 1.834%. With the fixation of 7000 iterations, the variants of the models with contrastive loss and binary cross entropy loss function have been built. The model with binary cross entropy loss function performed better with an improvement of 10.659%. With the fixation of the loss function as binary cross entropy, the number of classes in the support set has been varied from 10 and 20 classes. The variant model with ten classes in the support set performed better, with an improvement of 9.359%. Finally, the proposed model with 256 image pairs, 7000 iterations, binary cross entropy loss and a support set with 10 classes achieved an accuracy of 88.8%, which is shown in bold in the Table 2. Further, data in Omniglot has been augmented and its characters have been downscaled to achieve an accuracy of 92.8% for 10-way 1-shot.

With the same configuration, the proposed TNN model was tested for the Omniglot dataset. Further, these models were also tested for Tamil and MNIST datasets to observe the generalization of the built models. Handwritten characters of both the Tamil and MNIST datasets were resized into 64×64 pixels. Further, they were pre-processed with image dilation and thresholding. Figures 10 and 11 show the character before and after pre-processing. Same experiments have been conducted for the existing work of Koch et al.[3]for Omniglot, Tamil and MNIST datasets. Table 3 shows the performance comparison of the existing work, proposed SNN and TNN models for different datasets and it is inferred that the accuracy of Existing work, SNN and TNN performs better for 5-way 1-shot when compared to its counterparts. Further, it is also clear that for every dataset, TNN performs better when compared to Existing work and SNN. The performance of TNN over SNN for Omniglot is 2.2%. The performance of TNN over SNN for Tamil characters is 6.4%. The performance of TNN over SNN for MNIST is 12.2%. An improvement of 12.2% is seen in MNIST as all the 10 digits are distinct in its structure. In the Tamil character dataset, few characters are similar in structure. So there is relatively less improvement. In the Omniglot dataset, some of the characters across different languages are similar. So, there is less improvement. The performance comparison of existing work, proposed SNN and TNN models for different datasets are elucidated in Figures 13, 14 and 15.

Digitization of Handwritten English Characters

The built generalized model was tested for recognizing the characters from the handwritten English documents. A written document is scanned with a mobile camera under suitable illumination for clear processing of the image. The captured image as shown in Figure 12.a, may contain unwanted noise and it undergoes various stages of pre-processing. In pre-processing, the image is resized, converted into grayscale, and the noise is removed by dilation followed by erosion. Figures 12.b and 12.c show the gray scaled and cleaned image. The cleaned image is segmented into different logical parts, like lines of the paragraph, words of the line, and characters of the word as shown in

Figure 12.d. The segmented individual characters are shown in Figure 12.d and 12.e. A support set which consists of all 26 handwritten characters of English in lower case is generated. The segmented characters are recognized using the trained TNN and SNN models with the support set. An accuracy of 81.25% and 73.22% are obtained using TNN and SNN respectively. In addition, the characters are also recognized using Pytesseract, which is an open-source OCR engine. The performance of the digitization of the handwritten document has been tested using Pytesseract API, SNN, and TNN. For testing, four different documents were prepared with increased complexities in handwriting. The accuracies for each of the documents were measured and plotted as shown in Figure 13 and it is inferred that as the complexity of the handwriting increases, TNN performs better when compared to Pytesseract and SNN.

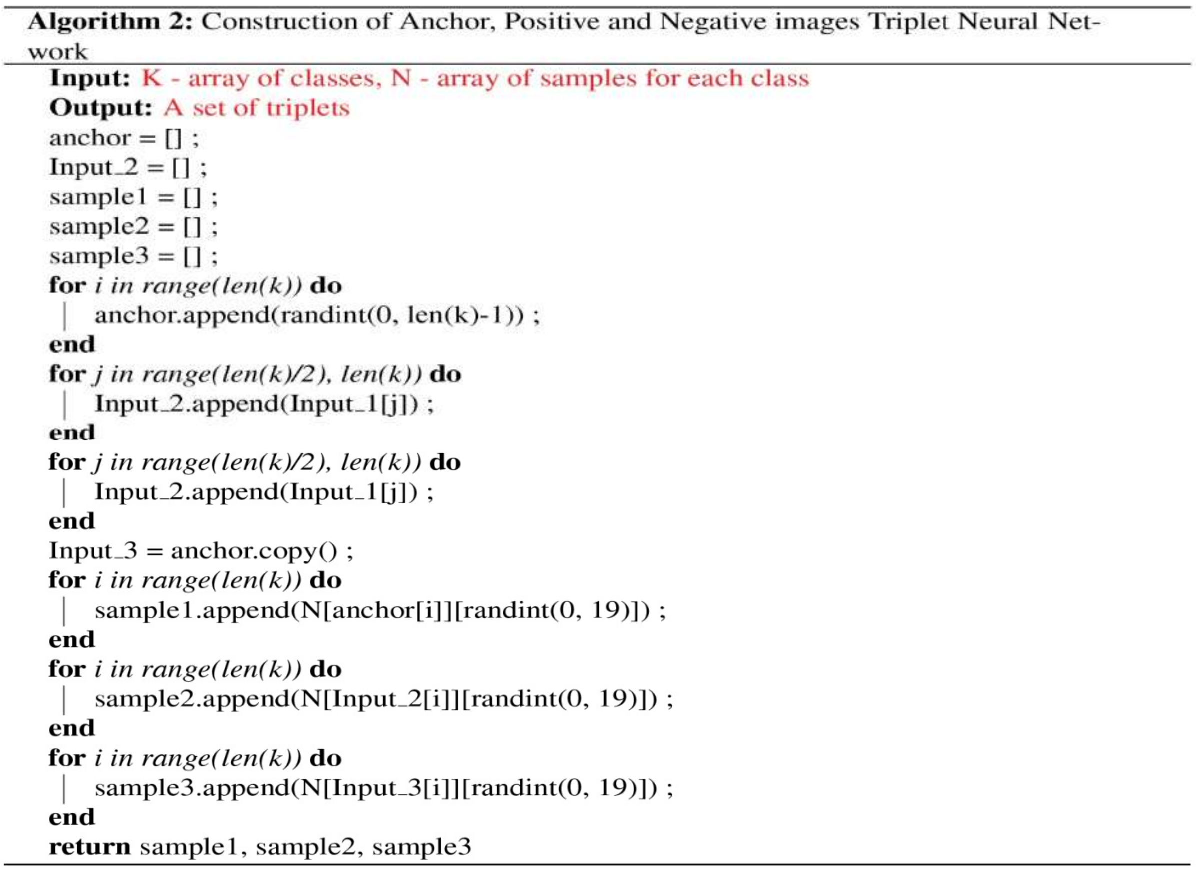
REFERENCES

1. Sparsh K., Nitesh P., and Vijaypal S.D. (2021). Handwritten Character Recognition using Deep Neural Networks. In proceedings of the International Conference on Data Science, Machine Learning and Artificial Intelligence. pp. 167–170. ACM.Pareek J., Singhania D., Kumari R. R., and Purohit S. (2020). Gujarati handwritten character recognition from text images. Procedia Computer Science. Vol. 171:514-523.
2. Sonawane P. K., and Shelke S. (2018). Handwritten Devanagari Character Classification using Deep Learning. In proceedings of International Conference on Information, Communication, Engineering and Technology (ICICET) pp. 1-4. IEEE.
3. Koch G., Zemel R., and Salakhutdinov R. (2015). Siamese neural networks for one-shot image recognition. ICML deep learning workshop. Vol. 2:10-16.
4. Wang B., and Wang D. (2019). Plant leaves classification: A few-shot learning method based on siamese network. IEEE Access. Vol. 7:151754-151763.
5. Argueso D., Picon A., Irusta U., Medela A., San-Emeterio M. G., Bereciartua A., Alvarez-Gila A. (2020). Few-Shot Learning approach for plant disease classification using images taken in the field. Computers and Electronics in Agriculture Journal. Vol. 175:650-657.
6. Devi P. R., Yashashvini R., Navyadhara G., and Ruchitha M. (2021). Similar Face Detection for Indian Faces using Siamese Neural Networks. In proceedings of 2nd International Conference for Emerging Technology. pp. 1-5. IEEE.
7. Heidari M. and Fouladi-Ghaleh K. (2020). Using Siamese Networks with Transfer Learning for Face Recognition on Small-Samples Datasets. In proceedings of International Conference on Machine Vision and Image Processing. pp. 1-4. IEEE.
8. Marattukalam F., Abdulla W. H., and Swain A. (2021). N-shot Palm Vein Verification Using Siamese Networks. In proceedings of International Conference of the Biometrics Special Interest Group. pp. 1-5. IEEE.
9. Kim M. S., Park T., and Park P. (2019). Classification of steel surface defect using convolutional neural network with few images. In proceedings of 12th Asian Control Conference. pp. 1398-1401. IEEE.
10. Balaha H. M., Ali H. A., Saraya M., and Badawy M. (2021). A new Arabic handwritten character recognition deep learning system (AHCRDLS). Neural Computing and Applications. Vol. 33(11):6325-6367.
11. Ashiquzzaman A., Tushar A. K., Rahman A., and Mohsin F. (2019). An efficient recognition method for handwritten arabic numerals using CNN with data augmentation and dropout. In proceedings of Data management, analytics and innovation. pp. 299-309. Springer.
12. Li H., Ren X., and Lv Y. (2019). One-shot chinese character recognition based on deep siamese networks. In proceedings of Chinese Intelligent Systems Conference. pp. 742-750. Springer.
13. Ahlawat S., Choudhary A., Nayyar A., Singh S., and Yoon B. (2020). Improved handwritten digit recognition using convolutional neural networks (CNN). Sensors. Vol. 12:3344-3350.
14. Naidu D. S., and Rafi T. M. (2021). Handwritten Character Recognition using Convolutional Neural Networks. International Journal of Computer Science and Mobile Computing. Vol. 10(8):41–45.

15. Bai J., Chen Z., Feng B., and Xu B. (2014). Image character recognition using deep convolutional neural network learned from different languages. In proceedings of International Conference on Image Processing. pp. 2560–2564. IEEE.
16. Chakrapani G. V. A., Chanda S., Pal U., and Doermann D. (2019). One-shot learning-based handwritten word recognition. In proceedings of Asian Conference on Pattern Recognition. pp. 210-223. Springer.
17. Mukhtar H., Khan M. Z., Khan M. U., and Younis H. (2021). Wheat Disease Recognition through One-shot Learning using Fields Images. In proceedings of International Conference on Artificial Intelligence. Pp. 229-233. IEEE.

Algorithm 1: Construction of Similar and Dissimilar pairs for training the Siamese Neural Network

```
Input: K - array of classes, N - array of samples for each class
Output: A set of similar and dissimilar pairs along with associated targets (0: dissimilar,
        1: similar)
targets = [ -1 for i in range(len(k))] ;
for i in range(len(k)) do
    if i >= len(k)/2 then
        targets[i] = 1 ;
    else
        targets[i] = 0 ;
    end
end
Input_1 = [] ;
Input_2 = [] ;
sample1= [] ;
sample2= [] ;
for i in range(len(k)) do
    Input_1.append(randint(0, len(k)-1)) ;
end
for j in range(len(k)/2), len(k)) do
    Input_2.append(Input_1[j]) ;
end
for j in range(len(k)/2), len(k)) do
    Input_2.append(Input_1[j]) ;
end
for i in range(len(k)) do
    sample1.append(N[Input_1[i]][randint(0, 19)]) ;
end
for i in range(len(k)) do
    sample2.append(N[Input_2[i]][randint(0, 19)]) ;
end
return sample1, sample2, targets
```



Algorithm 2: Construction of Anchor, Positive and Negative images Triplet Neural Network

```
Input: K - array of classes, N - array of samples for each class
Output: A set of triplets
anchor = [] ;
Input_2 = [] ;
sample1 = [] ;
sample2 = [] ;
sample3 = [] ;
for i in range(len(k)) do
    anchor.append(randint(0, len(k)-1)) ;
end
for j in range(len(k)/2), len(k)) do
    Input_2.append(Input_1[j]) ;
end
for j in range(len(k)/2), len(k)) do
    Input_2.append(Input_1[j]) ;
end
Input_3 = anchor.copy() ;
for i in range(len(k)) do
    sample1.append(N[anchor[i]][randint(0, 19)]) ;
end
for i in range(len(k)) do
    sample2.append(N[Input_2[i]][randint(0, 19)]) ;
end
for i in range(len(k)) do
    sample3.append(N[Input_3[i]][randint(0, 19)]) ;
end
return sample1, sample2, sample3
```

Table 1. Performance of variants of SNN Models

Loss Function	Interation	Batch size	Training Accuracy	Test Accuracy 20-way 1-shot	Test Accuracy 10-way 1- shot
Binary Cross Entropy	3000	128	78.8%	71.8%	86%
		256	86.4%	74.4%	87.2%
	7000	128	82.8%	80.8%	86%
		256	90%	81.2%	88.8%
Contrastive loss	3000	128	56.4%	51.6%	60.8%
		256	76.0%	78.8%	78.8%

Table 2. Inference on variants of SNN Models

S.No	Change factor	%improvement in Accuracy
1	Model trained with 256 pairs over 128 pair	3.2%
2	Model built with 7000 iterations over 3000 iterations	1.834%
3	Model built with binary cross entropy over contrastive loss	10.659%
4	Model tested with 10-way one shot over 20-way one shot	9.359%

Table 3. Performance of Existing (Koch et al), Proposed SNN and TNN models

Dataset	Evaluation Metric	Models		
		Existing work	SNN	TNN
Omniglot	Test Accuracy 5-way 1-shot	86%	94%	**96%**
	Test Accuracy 10-way 1-shot	84.9%	92.8%	93%
	Test Accuracy 20-way 1-shot	76.8%	84%	86%
Tamil	Test Accuracy 5-way 1-shot	80%	88%	**93.6%**
	Test Accuracy 10-way 1-shot	76%	83.6%	88.8%
	Test Accuracy 20-way 1-shot	68%	74.8%	80%
MNIST	Test Accuracy 5-way 1-shot	57.2%	64.4%	**73.6%**
	Test Accuracy 10-way 1-shot	49%	55.2%	62.8%
	Test Accuracy 20-way 1-shot	43.8%	49.3%	56.5%

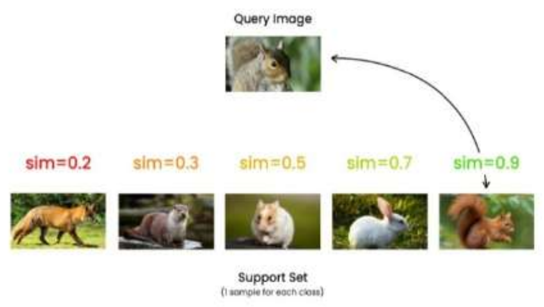


Figure 1. Identification of class of the query image in Few-Shot Learning

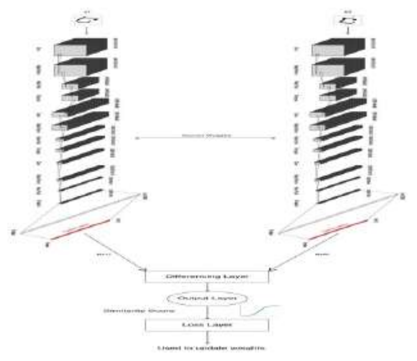
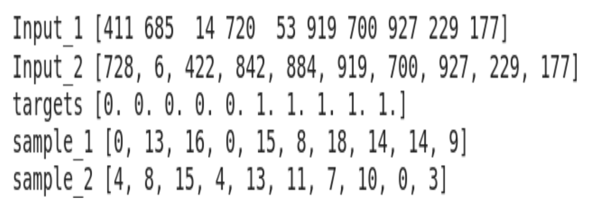


Figure 2. Design of Proposed Siamese Neural Network



```
Input_1 [411 685  14 720  53 919 700 927 229 177]
Input_2 [728, 6, 422, 842, 884, 919, 700, 927, 229, 177]
targets [0. 0. 0. 0. 0. 1. 1. 1. 1. 1.]
sample_1 [0, 13, 16, 0, 15, 8, 18, 14, 14, 9]
sample_2 [4, 8, 15, 4, 13, 11, 7, 10, 0, 3]
```

Figure 3. Construction of similar and dissimilar pairs

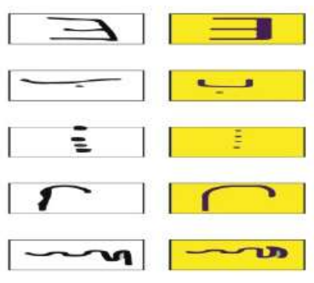


Figure 4. Generated similar pairs of characters

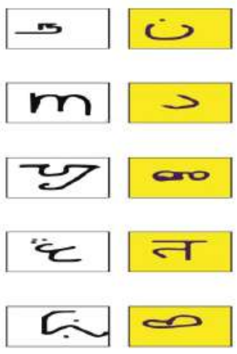


Figure 5. Generated dissimilar pairs of characters

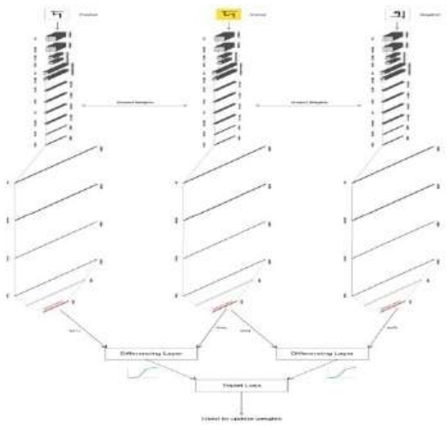
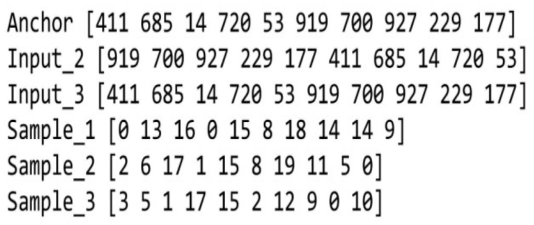


Figure 6. Design of Proposed Triplet Neural Network


```
Anchor [411 685 14 720 53 919 700 927 229 177]
Input_2 [919 700 927 229 177 411 685 14 720 53]
Input_3 [411 685 14 720 53 919 700 927 229 177]
Sample_1 [0 13 16 0 15 8 18 14 14 9]
Sample_2 [2 6 17 1 15 8 19 11 5 0]
Sample_3 [3 5 1 17 15 2 12 9 0 10]
```
Figure 7. Construction of triplets

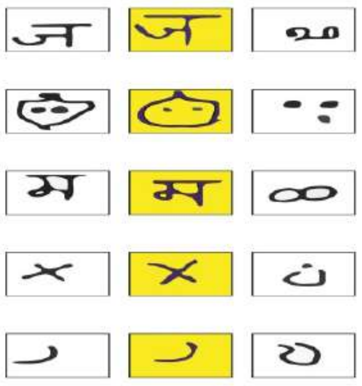


Figure 8. Generated triplets

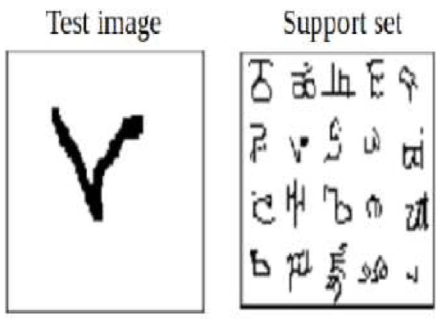


Figure 9. Test image and Support set

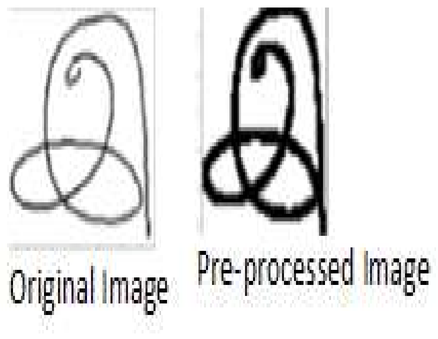


Figure 10. Original and Pre-processed images of Tamil dataset

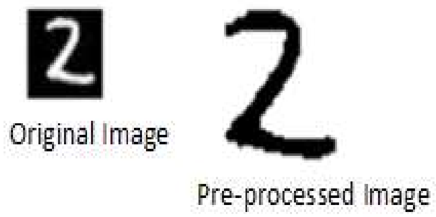
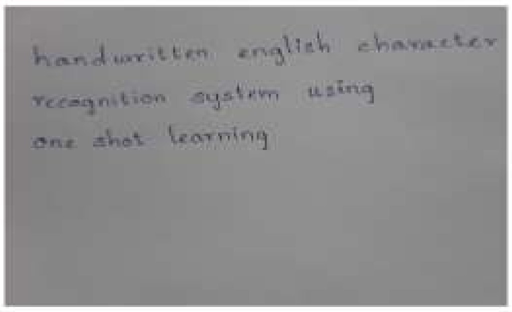
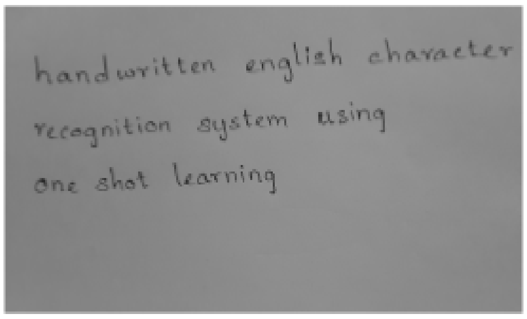


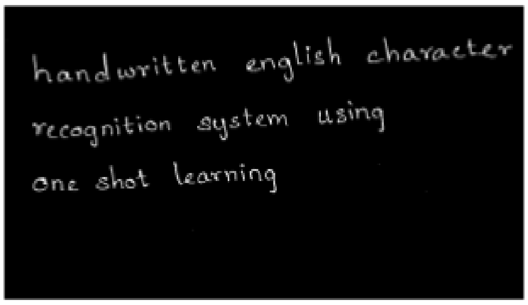
Figure 11. Original and Pre-processed images of MNIST dataset



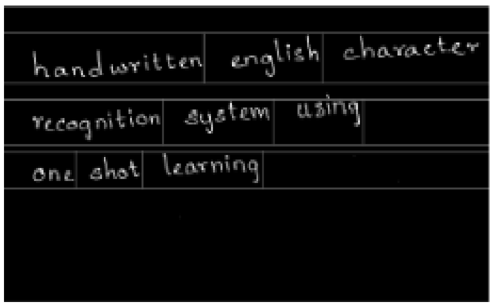
A. Handwritten English Text



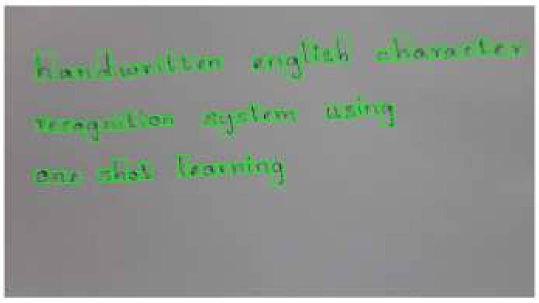
B. Gray scale Image



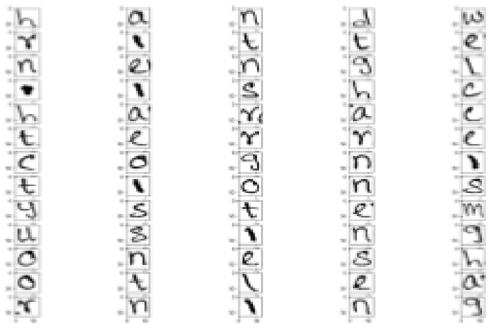
C. Image after binarization and noise removal



D. Image after line and word segmentation



E. Image after character detection



F. Individual segmented characters

Figure 12. Recognition of English characters

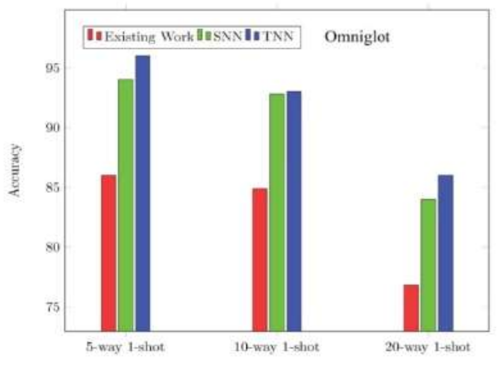


Figure 13. Performance of Existing Work, SNN, TNN for Omniglot Dataset

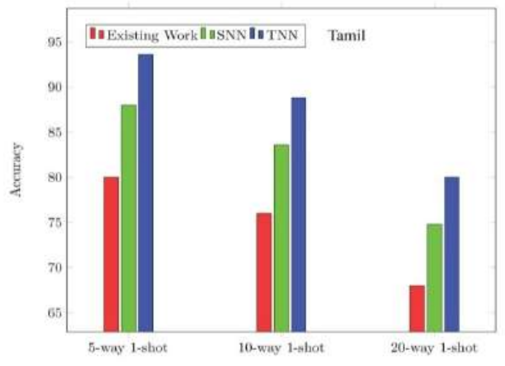


Figure 14. Performance of Existing Work, SNN, TNN for Tamil Dataset

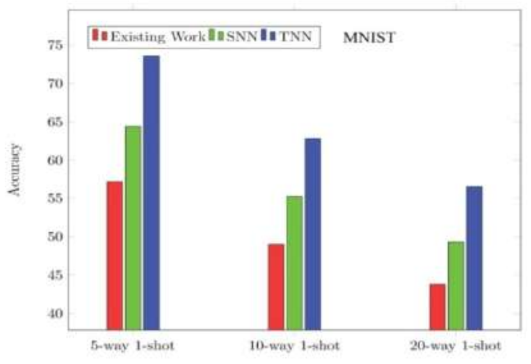


Figure 15. Performance of Existing Work, SNN, TNN for MNIST Dataset

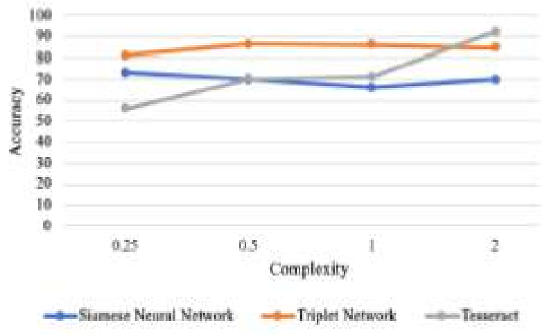


Figure 16. Performance comparison of OCR for English Text with the proposed models