One-Shot Learning using Siamese Network

Preface :

The implementation of this model has been carried on by Nikita Chauhan. The model has been completed but the performance of the model is not good.

Approach :

I have trained the model on a subset of the Omniglot dataset, it is one of the most popular datasets used for one-shot learning. It is specially designed to compare and contrast the learning abilities of humans and machines. This dataset consists of handwritten characters of 50 languages (alphabets) with 1623 total characters. There are only 20 samples for each character, each drawn by a distinct individual. The dataset is divided into 2 sets: background set and evaluation set. The background set contains 30 alphabets (964 characters) and is used to train the model whereas the remaining 20 alphabets are for pure evaluation purposes only.

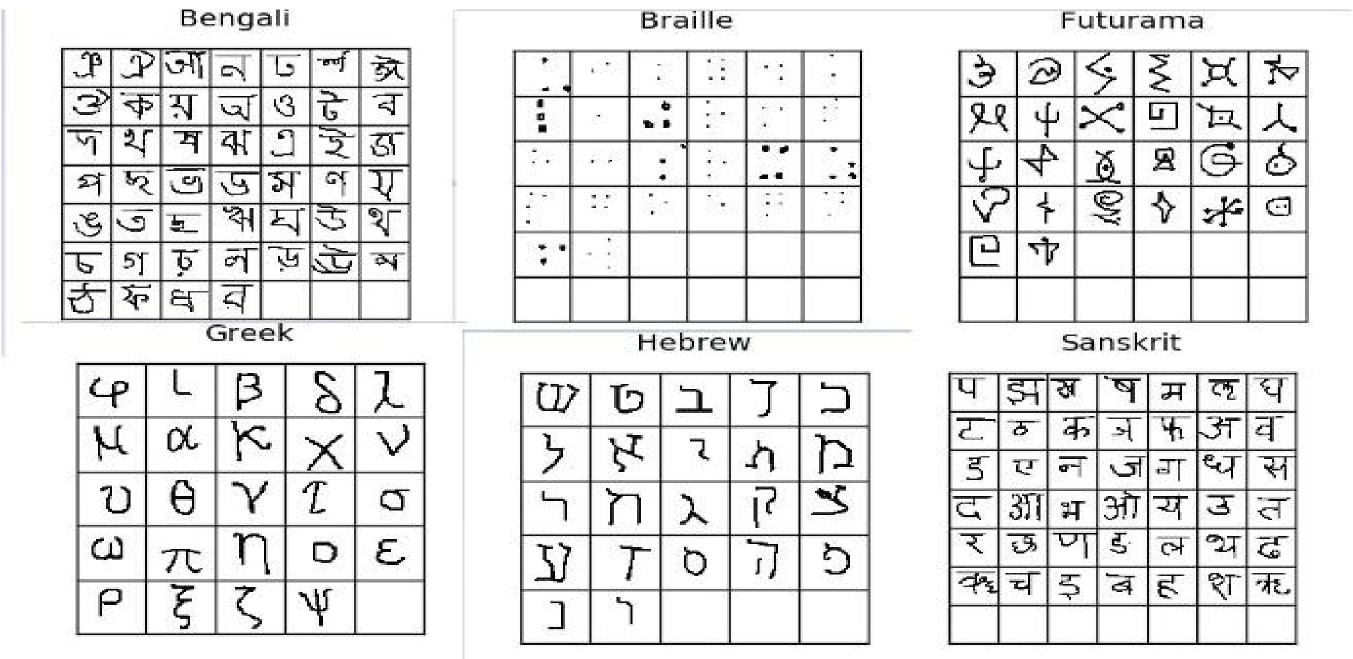

Fig1. The Omniglot dataset contains a variety of different images from alphabets across the world.

To develop a model for one-shot image classification, I have implemented a neural network that can discriminate between the class-identity of image pairs, which is the standard verification task for image recognition. The verification model learns to identify input pairs according to the probability that they belong to the same class or different classes. This model can then be used to evaluate new images, exactly one per novel class, in a pairwise manner against the test image.

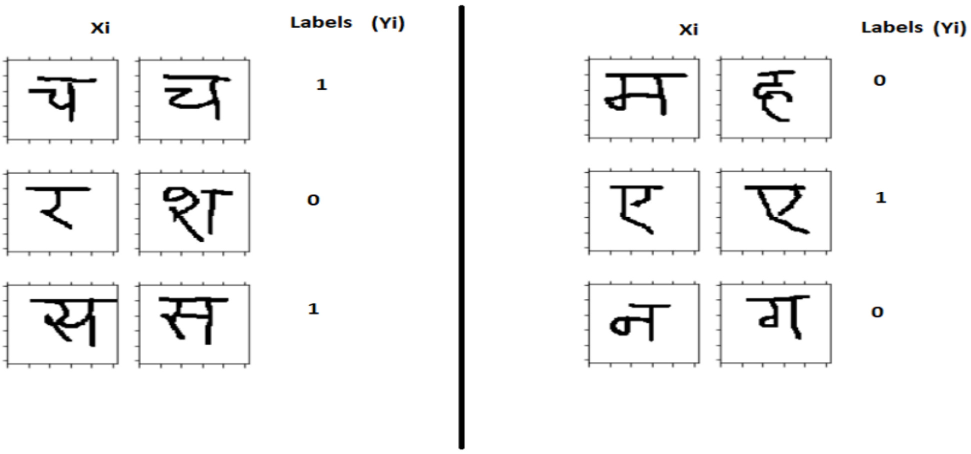

Fig2.Sample of 6 data points

The pairing with the highest score according to the verification network is then awarded the highest probability for the one-shot task. First, I have trained it on the training set so that the verification model can learn the features to become sufficient to confirm or deny the identity of characters from one set of alphabets, then it ought to be sufficient for other alphabets, provided that the model has been exposed to a variety of alphabets to encourage variance amongst the learned features.

Model Architecture and Training:

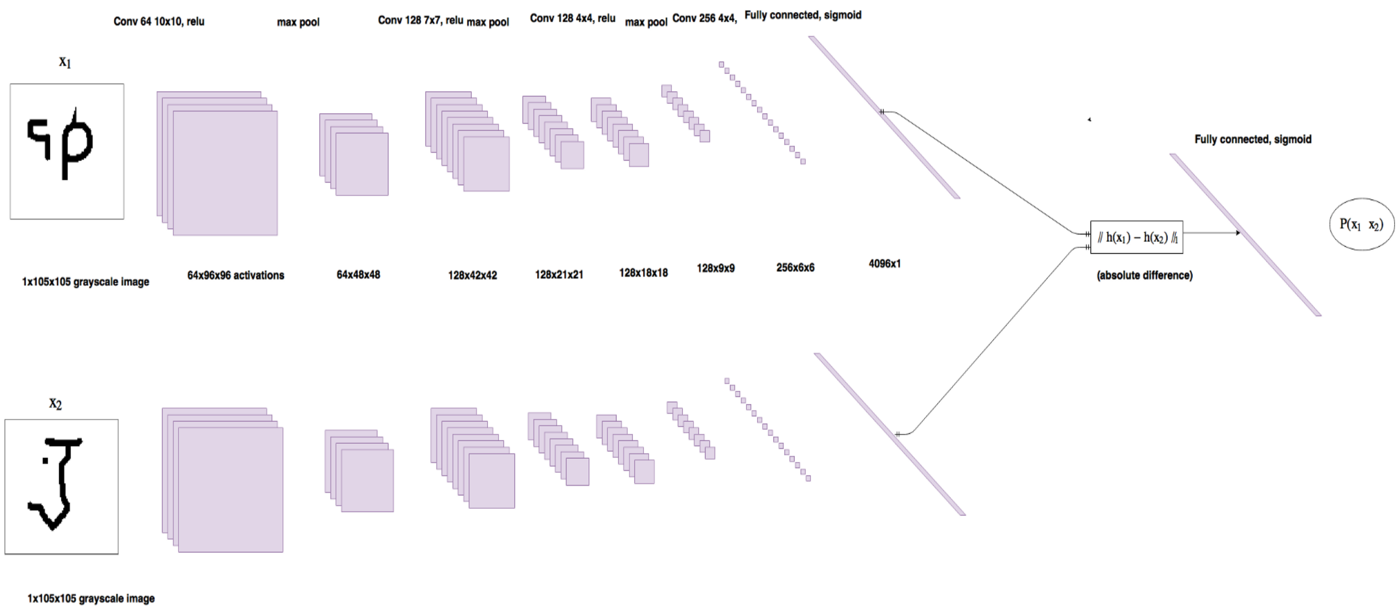

Fig3. A high-level architecture

For this work, we first implemented the basic architecture of Siamese Networks and Matching Networks using the sigmoid Activation function. The Siamese network is an architecture with two parallel layers.

In this architecture, instead of a model learning to classify its inputs using classification loss functions, the model learns to differentiate between two given inputs. It compares two inputs based on a similarity metric and checks whether they are the same or not. Similar to any deep learning architecture, a Siamese network also has two phases: Training and Testing Phase.

But usually, for a One-shot learning approach (as we won't have a lot of data points) we train the model architecture on a different dataset and test it for our less amount of dataset.

For the feature extraction, I have used transfer learning(i.e pre-trained network on imagenet), which allows the network to train faster as it doesn't learn from scratch. And the units in the final convolutional layer are flattened into a single vector. The vgg16 is followed by a fully connected layer, and then one more layer computing the induced distance metric between each Siamese twin, which is given to a single sigmoidal output unit.

After optimizing the siamese network to master the verification task, then I tried to demonstrate the discriminative potential of our learned features at one-shot learning.

To empirically evaluate one-shot learning performance, Lake developed a 20-way within-alphabet classification task in which an alphabet is first chosen from among those reserved for the evaluation set, along with twenty characters taken uniformly at random. Two of the twenty drawers are also selected from among the pool of evaluation drawers. These two drawers then produce a sample of the twenty characters. Each one of the characters produced by the first drawer is denoted as test images and individually compared against all twenty characters from the second drawer, with the goal of predicting the class corresponding to the test image from among all of the second drawer's characters. An individual example of a one-shot learning trial is depicted in Figure 4.

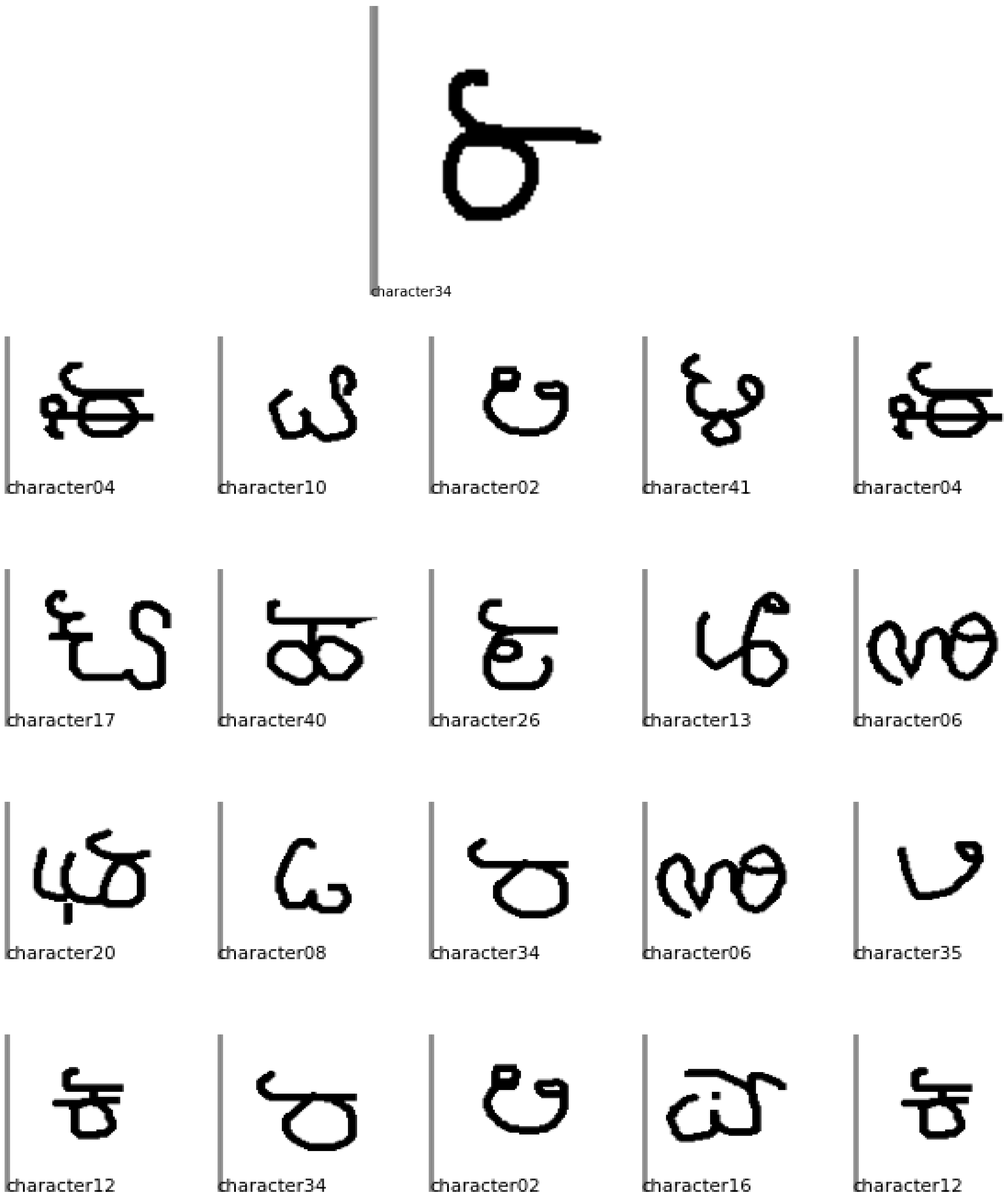

Fig.4

Conclusion:

I have trained the model using a TensorFlow backend in Keras. I have not been able to reproduce the results reported by the authors (>90% in the evaluation set). I was able to get results of around 60%.

Now I'm working on how to improve the performance of the model.