SANJAY JARAS

<https://sanjayjaras.github.io/>

Devanagari Handwritten Character Classification

2021

BELlevue university

**Abstract**

Devanagari script also called Nagari script is mostly used in the Indian subcontinent. This script is based on the ancient Brahmi script. Devanagari script is the most used Indian script. The Devanagari script is composed of 47 primary characters including 14 vowels, 33 consonants, and 10 digits. It is the fourth most widely adopted writing system in the world being used for over 120 languages. Some of the languages using Devanagari script as a writing system are Sanskrit, Hindi, Marathi, Pali, Kashmiri, and Nepali. This script is used by more than 500 million people worldwide. The Devanagari script has consonants and modifiers however for this project I am considering the dataset with consonants. In the past few years, deep neural networks are getting highly used in computer vision and machine learning tasks such as regression, segmentation, classification, detection, pattern recognition, etc. because of their outstanding performance. Machine learning can help us in digitizing different handwritten documents in the Devanagari script. This can help us better transfer and store those documents.

**Intro/background of the problem**

In the United States, various banks support submitting checks to banks through mobile applications by taking photographs. These checks are written in English and banks are using applications to read these checks. Similarly, the address written in English script on letters can be read with computer vision and as per addresses they can be classified. Every character is written with several variations while writing the Devanagari as a writing system. Devanagari and in turn Hindi is the national language of India and Nepali is the national language of Nepal and adopted by more than 500 million people worldwide. Contrary to wide adaptation, not much of the work happened with the Devanagari script as compared to other scripts like English. The handwritten character recognition will help in preserving the ancient document written in the Devanagari script. There are other applications of offline handwriting recognition(explained more about this in appendix section) like reading postal addresses, bank check amounts, and forms. Creating digital libraries, allowing the entry of image textual information into computers by digitization, image restoration, and recognition methods. Convolution neural networks are very good when analyzing images so, I have used Convolution neural network model for my final model.

**Methods**

For this project, the dataset is obtained from Machine Learning Repository ([Center for Machine Learning and Intelligent Systems](http://cml.ics.uci.edu/))[1]. This is an image database of Handwritten Devanagari characters. There are 46 classes of characters with 2000 examples of each character. The dataset is split into a training set (85%) and a testing set (15%). In total, there are 92000 images. The training dataset has 78200 images, and the Test dataset has 13800 images. The training dataset contains 1700 images for each character and the training dataset contains 300 images for each character. Each image is of 32x32 pixels with one channel. The actual character is centered within 28 by 28 pixels and padding of 2 pixels is added on all four sides of the actual character. The 46 classes contain 10 classes for digits and 36 for other characters. When we want to do image classification, each image should have one or more labels assigned to it and the prediction model needs to predict a label for each image. The images are stored in the folder structure split into Train and Test. Under these folders, there is one folder for each character. The folder name indicates the character name(class). Each folder from the Train folder contains 1700 image files. Similarly, folders under the Test folder have 300 files under each folder. I read those all images in the NumPy array and the folder name became the label of each of the images.

I tried two different kinds of models for this project first by using machine learning algorithms with scikit-learn and second deep learning model using TensorFlow. For machine learning models the dataset is divided into training and test set as I have used cross-validation for model validations. For these algorithms, I created two data frames one for input images and the other for labels. I created two transformer classes one for normalizing the image data to 0 through 1 and the second label encoding the labels. Created pipeline to preprocess data using the above two classes. For finding a better algorithm and parameters, I used grid search. For grid search, I selected the following algorithms KNeighborsClassifier, DecisionTreeClassifier, RandomForestClassifier, LGBMClassifier, SVM. I couldn’t use GradientBoostingClassifier and XGBClassifier due to hardware limitations. For grid search, I used only 100 images out of 1700 of each character and used cross-validation by using RepeatedStratifiedKFold with 2 folds. For these models, I didn’t use image augmentation as these models were taking too long to train.

For the second approach using deep learning, I used Tensorflow to create CNN(convolution neural network) model. The images are read into NumPy arrays of shape(32,32,1). The original image only has 32x32 pixels however, I added one extra dimension to work with the Tensorflow model. I double the training image dataset size by adding augmented images. For augmentation, I used random rotation between 1-10 degrees. I used a small rotation angle as if an image rotated with a bigger angle might create a different or invalid character. For label encoding, I used Keras StringLookup function. This function helps in converting labels to int and back to label by using vocabulary. The train and test dataset is then loaded as a tensor dataset to generate a random batch for each training epoch. The test dataset is then split 50-50 into test and validation datasets. Then the CNN model is created by using 11 layers. The first layer was added to normalize the image data to value through 0-1. I used 2 dense layers one with 256 units and the last dense layer with 46 classes as output. A dropout layer is added to avoid the overfitting of models. The model is then compiled and trained with a batch size of 32.

**Results**

Machine Learning models:The first approach with machine learning models yields accuracy around 40% to 82% on the validation dataset while training. The Decision Tree is being the worst with 40% while the SVM is the best with 82% accuracy on the training dataset.

**Chart, bar chart

Description automatically generated**

Then SVM model is finally trained with full training data and best parameters found in grid search as it yields the best results. When this model was tested with a test dataset it shows an accuracy of 62%. This shows that the model is over-fitted and needs more work.

**Deep Learning Model:** With the second approach of the deep learning model, the model consistently produced good results on the validation dataset while training. The validation dataset accuracy was around 96% in the first epoch and around 99.27% on the last epoch. The following plots show the accuracy and loss comparison of the model on the training and validation dataset. When this model is tested with the test dataset, it yields an accuracy of 99%(98.83%). This shows that the model is not over-fitted and it is working very well on unseen data.

Chart, histogram

Description automatically generated

**Discussion/conclusion – Next steps**

The model is doing quite well on unseen data, so we can say that this model is ready for the real world. On Devanagari Handwritten Character Dataset Data Set[1] it is mentioned that they have created a model that can yield accuracy up to 98.47%. That model is explained in the [paper.](https://ieeexplore.ieee.org/document/7400041)  I am happy that the model can surpass the accuracy(98.83%) achieved by that model accuracy(98.47%). I think this model could do that because of augmented images addition in the training dataset.

**Acknowledgments**

The author of this project has referred to the Center for Machine Learning and Intelligent Systems[1] for the data set. The author referred to some notebooks available on Kaggle and GitHub. There is a significant number of solutions available on the Internet for the problem. The author has referred to some of those available solutions and has designed my solution. The author has also referred to multiple websites(not limited to the below references) including data science articles from various authors, machine learning websites, mediam.com, towardsdatascience.com, and many others for basic machine learning and statistics concepts and practical examples.

**References**

1. Devanagari Handwritten Character Dataset Data Set - Center for Machine Learning and Intelligent Systems <https://archive.ics.uci.edu/ml/datasets/Devanagari+Handwritten+Character+Dataset>
2. Classifier comparison on Devanagari recognition - Prashanth Aditya Susarla - <https://www.kaggle.com/spadix/classifier-comparison-on-devanagari-recognition>
3. Handwritten Devanagari character identification with ResNet - Akshay Chougule - <https://medium.com/@akshaychougule/handwritten-devanagari-character-identification-using-resnet-b90894b42c4d>
4. Devanagari - <https://en.wikipedia.org/wiki/Devanagari>
5. Deep Learning based Character Classification using Synthetic Dataset - Krutika Bapat - <https://learnopencv.com/deep-learning-character-classification-using-synthetic-dataset/>
6. Devanagari Character Set - Rishi Anand - <https://www.kaggle.com/rishianand/devanagari-character-set>
7. Intro to Computer Vision-Indian National Language - Shruti Iyyer - <https://www.kaggle.com/shrutimechlearn/intro-to-computer-vision-indian-national-language>
8. Devanagari Character Dataset - Ashok Pant - <https://www.kaggle.com/ashokpant/devanagari-character-dataset>
9. Image classification - <https://www.tensorflow.org/tutorials/images/classification>
10. Recognizing Handwritten Digits with Scikit-learn -
11. Manthan Bhikadiya - <https://medium.com/codex/recognizing-handwritten-digits-with-scikit-learn-90ca6e2471ed>
12. Handwriting recognition - <https://en.wikipedia.org/wiki/Handwriting_recognition>
13. An Analysis of Irregularities in Devanagari Script Writing – A Machine Recognition Perspective - Satish Kumar - <http://www.enggjournals.com/ijcse/doc/IJCSE10-02-02-30.pdf>

**Appendix**

Offline handwritten recognition: This involves the conversion of already written text into digital information. In this method, the letters are available in images; and these letters are then converted to label codes usable by a computer. The input data in this process is static data like photocopies of old handwritten documents. There are different techniques used for offline handwritten recognition like character extraction, character recognition, feature extraction, and modern techniques like machine learning and deep learning. In this project, I covered both of these modern techniques.

Online handwritten recognition: This technique involves the conversion of text as it is written on special digital pads or smart-device screens. The sensors beneath the screens/pads capture the pen/stylus movement and convert it into digital information. Then this information can be used by computers. It requires both specialized hardware like stylus touch-sensitive screens and software.