Capstone Propasal
Handwritten Devnagri Character & Numeral Recognition using Deep Learning

Nisha Gadhe
12th April 2018

Handwritten Devnagari Characters & Numerals Recognization

[Proposal & The domain background — the field of research where the project is derived;]

Handwriting recognition (or HWR) is the ability of a computer to receive and interpret intelligible handwritten input from sources such as paper documents, photographs, touch-screens and other devices. The image of the written text may be sensed "off line" from a piece of paper by optical scanning (optical character recognition) or intelligent word recognition. Alternatively, the movements of the pen tip may be sensed "on line", for example by a pen-based computer screen surface, a generally easier task as there are more clues available.

Nowadays, techniques such as deep neural networks are sucessfully used on devanagari characters & digits. We propose here to apply some of these techniques to the devanagari characters, used to write in India and Nepal.

Devanagari is part of the Brahmic family of scripts of Nepal, India, Tibet, and South-East Asia. The script is used to write Nepali, Hindi, Marathi and similar other languages of South and East Asia. The Nepalese writing system adopted from Devanagari script consists of 12 vowels, 36 base forms of consonant, 10 numeral characters and some special characters. Vowel characters are shown in Fig. 1, consonants characters in Fig. 2 and numeral characters in Fig. 3. Moreover, all 36 consonants could be wrapped with the vowels generating 12 other derived forms for each branch of consonant character.

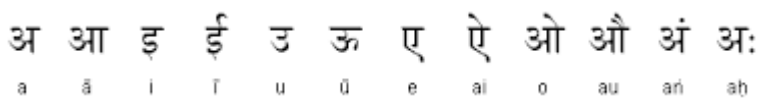


Fig. 1 vowels

अ	आ	इ	ई	उ	ऊ	ऋ	ए	ऐ	ओ
औ	क	ख	ग	घ	ङ	च	छ	ज	झ
ञ	ट	ठ	ड	ढ	ण	त	थ	द	ध
न	प	फ	ब	भ	म	य	र	ल	व
श	ष	स	ह						

(b) Devanagari

Fig. 2 Consonants

१	२	३	४	५	६	७	८	९	०
1	2	3	4	5	6	7	8	9	0

Fig. 3 Numeral

Problem Statement

[a problem being investigated for which a solution will be defined;]

The goal is to predict the handwritten devanagari Character in an image. The dataset contains images of handwritten devnagri characters which can be used to train and test the model.

The datasets and inputs

[data or inputs being used for the problem;]

This dataset contain Devanagari Characters. It comprises of 92000 images [32x32 px] corresponding to 46 characters, consonants "ka" to "gya", and the digits 0 to 9. The vowels are missing.

The CSV file is of the dimension 92000 * 1025. There are 1024 input features of pixel values in grayscale (0 to 255). The column "character" represents the Devanagari Character Name corresponding to each image.

There are 46 classes present out of which 10 classes for digits & 36 classes for characters & each class has 2000 images.

Please check below image of my desktop

```
In [24]: dataset.groupby("character").count()
```

Out[24]:

character	pixel_0000	pixel_0001	pixel_0002	pixel_0003	pixel_0004	pixel_0005	pixel_0006	pixel_0007	pixel_0008	pixel_0009	...	pixel_101
character_01_ka	2000	2000	2000	2000	2000	2000	2000	2000	2000	2000	...	200
character_02_kha	2000	2000	2000	2000	2000	2000	2000	2000	2000	2000	...	200
character_03_ga	2000	2000	2000	2000	2000	2000	2000	2000	2000	2000	...	200
character_04_gha	2000	2000	2000	2000	2000	2000	2000	2000	2000	2000	...	200
character_05_kna	2000	2000	2000	2000	2000	2000	2000	2000	2000	2000	...	200
character_06_cha	2000	2000	2000	2000	2000	2000	2000	2000	2000	2000	...	200
character_07_chha	2000	2000	2000	2000	2000	2000	2000	2000	2000	2000	...	200
character_08_ja	2000	2000	2000	2000	2000	2000	2000	2000	2000	2000	...	200
character_09_jha	2000	2000	2000	2000	2000	2000	2000	2000	2000	2000	...	200
character_10_yna	2000	2000	2000	2000	2000	2000	2000	2000	2000	2000	...	200
character_11_taamatar	2000	2000	2000	2000	2000	2000	2000	2000	2000	2000	...	200
character_12_thaa	2000	2000	2000	2000	2000	2000	2000	2000	2000	2000	...	200
character_13_daa	2000	2000	2000	2000	2000	2000	2000	2000	2000	2000	...	200
character_14_dhaa	2000	2000	2000	2000	2000	2000	2000	2000	2000	2000	...	200
character_15_adna	2000	2000	2000	2000	2000	2000	2000	2000	2000	2000	...	200
character_16_tabala	2000	2000	2000	2000	2000	2000	2000	2000	2000	2000	...	200
character_17_tha	2000	2000	2000	2000	2000	2000	2000	2000	2000	2000	...	200

From above image also there seems to be no class imbalanced in the dataset.

This Dataset has been extracted from Kaggle.

Data set Link : https://www.kaggle.com/rishianand/devanagari-character-set

Solution Statement

[a the solution proposed for the problem given]

Given that the problem is a supervised learning problem and more specifically a classificatio problem, thereare lot of algorithms available for training aclassifier to learn from the data. The algorithm chosen for this project is Deep Neural Network(DNN) using Convolutional Neural Network(ConvNet).

A Convolutional Neural Network (CNN)

CNN is a type of feed-forward Artificial Neural Network in which the connectivity pattern between its neurons is inspired by the organization of the animal visual cortex.

Convolutional Neural Networks consist of neurons that have learnable weights and biases. Each neuron receives some input, performs a dot product and optionally follows it with a non-linearity.

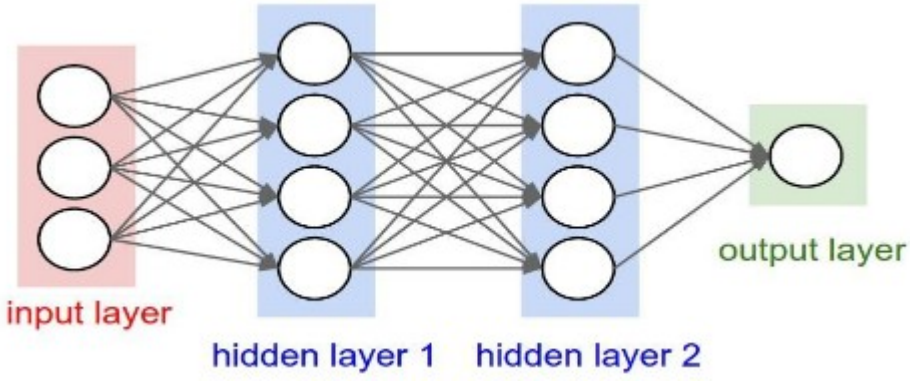


Fig. Convolutional Neural Network Basic Layout

A CNN consists of a lot of layers. These layers when used repeatedly, lead to a formation of a Deep Neural Network. Three main types of layers used to build a CNN are:

1) First Convolution Layer:

The convolution layer is the core building block of a convolutional neural network. It convolves the input image with a set of learnable filters or weights, each producing one feature map in the output image.

2) ReLU Function:

The Rectified Linear Unit apply an elementwise activation function, such as the max (0, x) thresholding at zero.

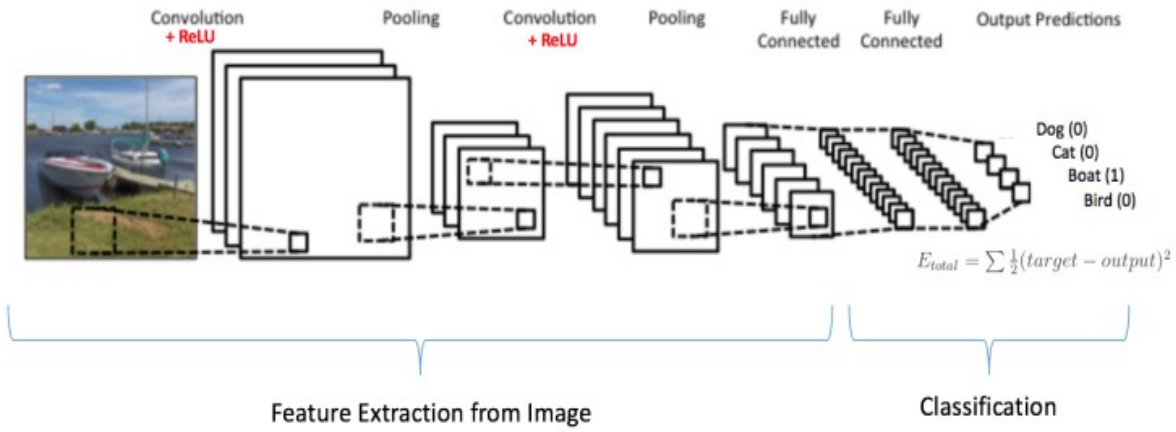
3) Pooling Layer:

The pooling layer is used to progres-sively reduce the spatial size of the representation to reduce the amount of parameters and computation in the network.The pooling layer takes small rectangular blocks from the convolution layer and subsamples it to produce a single output from that block. There are several ways to do this pooling, such as taking the average or the maximum, or a learned linear combination of the neurons in the block.

4) Fully Connected Layer:

The fully-connected layer is used for the high-level reasoning in the neural network. It takes all neurons in the previous layer and connects it to every single neuron it has. Their activations can be computed with a matrix multiplication followed by a bias offset as a standard neural networks.

An example of a Convolutional Neural Network is given below



Benchmark Model

[some simple or historical model or result to compare the defined solution to;]

The benchmark model solves the same handwritten devnagari charachers/Numeral recognition problem mentioned in this project by using CNN model. I am trying to achive more than 90% accuracy on test data. I do the model fitting on the whole training dataset with hyper-parameter 'number of epochs' =10. We can tune this parameter to achive better accuracy.

Evaluation Metrics

[functional representations for how the solution can be measured;]

After fitting the model, the model can be evaluated on the unseen test data. Using model.evaluate function in Keras, we can calculate loss value & accuracy value.

The model.evaluate() method computes the loss and any metric defined when compiling the model. Also I will check Model accuracy & loss with pyplot curves to ensure that this is not overfitting case.

Project Design

[how the solution will be developed and results obtained]

The dataset is loaded into python using keras library as discussed in the Dataset and Inputs section. The loaded data will be first explored and visualized using pandas, numpy and matplotlib library to understand the nature of the data.

Exploring the data will help us in deciding how to approach and whetherany preprocessing of the data is needed. Preprocessing of the data is done as required. Once the data is ready, the Deep neural network(DNN) will be built based on the architecture(ConvNet) as discussed in the Solution Statement. Once the model is built, it will be compiled to check if the architecture has any error.

Then the compiled model will be trained on the training data (X_train, y_train) and evaluated using accuracy score against the testing data (X_test, y_test).

Then the results can be analyzed and compared with respect to the benchmark model to know the overall performance of the model. Now we have a trained model, a test is conducted against the

model by loading images of devnagari characters/numeral which are not from downloaded Kaggle dataset to evaluate the performance of the model.

The images are loaded and then preprocessed to match the kaggel dataset format so that we can test it. The model is then made to predict the devnagari characters/numeral in the preprocessed image. Thus, we can evaluate our model’s performance over real time data.