**Project Report**

**Soft Computing**

**(ITE1015)**

**Improving OCR Accuracy**



SCHOOL OF INFORMATION AND TECHNOLOGY

AUGUST 2018

Submitted by:

Hariom Agrawal (16BIT0224)

Under the guidance of **Prof. VALARMATHI B – (SITE)**

**INDEX**

|  |  |  |
| --- | --- | --- |
| S.No | Description | Page |
| 1 | Abstract | 3 |
| 2 | Objective | 3 |
| 3 | Existing System | 4 |
| 4 | Proposed System | 4 |
| 5 | Dataset Description   1. Source Link 2. Dataset Description 3. Sample Dataset 4. Train and Test dataset | 5 |
| 6 | Literature Survey | 8 |
| 7 | Algorithm Description | 16 |
| 8 | Proposed Architecture | 18 |
| 9 | Code & Results – discussion   1. Code 2. Results (table) 3. Screenshots | 20 |
| 10 | Conclusion | 33 |
| 11 | Future Work | 33 |
| 12 | References | 34 |

**Abstract**

Optical Character Recognition (OCR) is a piece of software that converts printed text and images into digitized form such that it can be manipulated by machine. Unlike human brain which has the capability to very easily recognize the text/ characters from an image, machines are not intelligent enough to perceive the information available in image. Therefore, a large number of research efforts have been put forward that attempts to transform a document image to format understandable for machine. OCR is a complex problem because of the variety of languages, fonts and styles in which text can be written, and the complex rules of languages etc. Hence, techniques from different disciplines of computer science (i.e. image processing, pattern classification and natural language processing etc. are employed to address different challenges.

**Objective**

This project aims to develop a software which will detect the handwritten and machine text using proper algorithm and techniques of Soft Computing with neural networks. This project will also aim to develop a suitable and optimized algorithm for text detection. This project will also see for various different algorithms and will try to improve the efficiency of software. This project uses CNN algorithm with image processing using OpenCV in python for all the training and testing images before training it in neural system.

**Existing System**

In the running world there is a growing demand for the users to convert the printed documents in to electronic documents for maintaining the security of their data. Hence the basic OCR system was invented to convert the data available on papers in to computer processable documents, so that documents can be editable and reusable. The existing system of OCR deals with the basic data mining classification algorithm such as Support Vector Machines (SVM), and the accuracy comes out to be pretty low as around 80%. This system also uses some of the image processing for the pre-processing part.

**Proposed System**

Our proposed system is OCR which uses the existing system but we used neural networks to train our images instead of using classification algorithm. We used specifically CNN algorithm in our system as this algorithm is best for the dataset containing images. Making use of existing system helped us to remove the noise removal and smoothening and filtering the image and by using feature classification it helped us to take only that part of image which actually contains the image, hence it helped us to remove the unwanted part of images. This system can also be useful for the images of difference language as the requirement in this is only the training dataset, so this system also removes the barrier for the specific language. We used multiple characters ranging from lower-case a to z and also upper-case images from A to Z. This project also used the numbers from 0 to 9. This system is multi-lingual.

**Dataset Description**

1. **Source Link**

The dataset is named as char74K dataset and is taken from the below link:

<http://www.ee.surrey.ac.uk/CVSSP/demos/chars74k/EnglishFnt.tgz>

1. **Dataset Description**

The dataset comprises of:

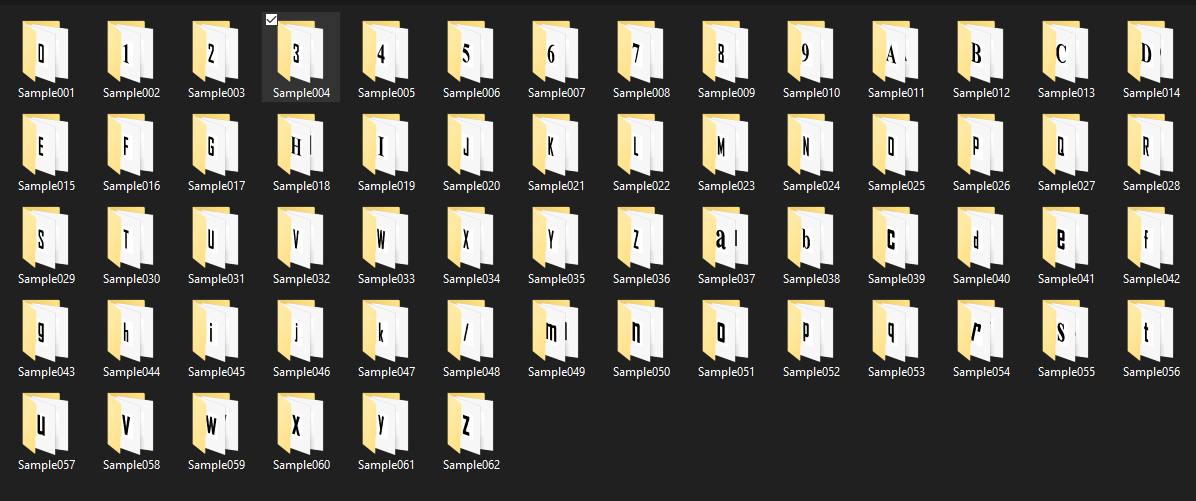
* 62 classes (0-9, A-Z, a-z)
* 7705 characters obtained from natural images
* 3410 hand drawn characters using a tablet PC
* 62992 synthesised characters from computer fonts

This gives a total of over 74K images (which explains the name of the dataset).

Each class consists of 1016 images.

The dataset does not come with the separate testing and training data.

1. **Sample Dataset**



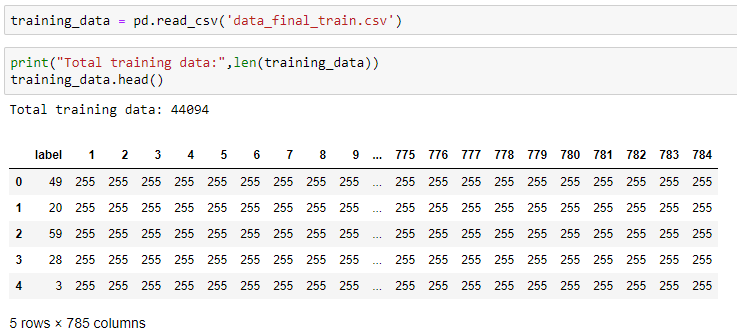
   

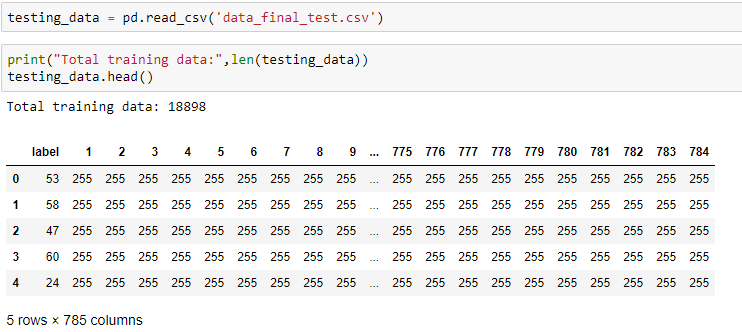
1. **Train and test Data**

The dataset was divided into 2 parts training and testing, training dataset comprised of 70% of total dataset i.e. the images for each of the classes were 711 which summed up to total of **44,082 images** and for testing the rest of the data was taken which summed up to a total of **18,910**. But before splitting the dataset we pre-processed it using openCv and then reduced the image size to 20X28 pixels. Then this image dataset was converted to csv file using some of the python module such as Pandas and NumPy and were shuffled thoroughly using scikit-learn module. Then it was divides into training and testing csv file.

**Training data:**



**Testing data:**



**Literature Survey**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Description** | **Algorithm** | **Dataset being used** | **Performance measures** | **Gap Identified** | **Proposed** |
| 1 | Author: Bharath V and other  Year: March 2018  A Font style classification system for English OCR. | Pre-Processing: Extracting each character  Step 2: Feature Computation  The combination of features computed from images include left distance profile, right distance profile and diagonal distance profile features. | Char74K dataset | Average accuracy of 80% | Poor accuracy | Improved by normalization of features and by using neural network such as CNN. |
| 2 | Improving classification using a Confidence Matrix based on weak classifiers applied to OCR.  By: Juan Ramon Rico-Juan, Jorge Calvo-Zaragoza | This paper proposes a new feature representation method based on the construction of a Confidence Matrix (CM). This representation consists of posterior probability values provided by several weak classifiers, each one trained and used in different sets of features from the original sample. The CM allows the final classifier to abstract itself from discovering underlying groups of features.  If Ω is a set of class labels and D is the set of weak classifiers, then a |D|X|Ω| matrix is obtained. | NIST SPECIAL DATABASE 3,  MNIST, etc. | It was observed that the average error for each dataset dropped by significant amount. | The main drawback is that it requires the user to be an active part of the process by extracting these different sets of features. | A bagging system could be explored in order to obtain different weak classifiers trained with a different subset of features. |
| 3 | Optical character recognition in real environments using neural networks and k-nearest neighbour.  By: Oliviu Matei and other. | Method combines two algorithms an artificial neural network on one hand, and the k-nearest neighbor as the confirmation algorithm.  neural network is a multi-layer perceptron (MLP) | Set of characters. | accuracy of 99.58% | Less dataset, less accuracy | Involving more datasets. |
| 4 | Author: Chowdhury Md Mizan, Tridib Chakraborty and Suparna Karmakar  Year: June 2017  Paper: International Journal of Advanced Research in Computer Science  ISSN No. 0976-5697 | Start  Scan the textual image.  Convert color image into gray image and then binary image.  Do preprocessing like noise removal, skew correction etc.  Load the DATABASE.  Do segmentation by separating lines from textual image. | Not Specified | Extremely efficient | Cannot detect some characters because of dataset. | Using of dictionary to find the neighbouring word. |
| 5 | Author: Ranjan Jana and other  Paper: International Journal of Computer Applications Technology and Research  ISSN No. 2319–8656 | STEP 1: The input text image is converted into binary image. STEP 2: The binary image is complimented so that the letters constitute by binary 1 (one) and background constitute by binary 0 (zero). STEP 3: All text lines are separated from the binary image. This is done by finding the sum of all values in a row. When the sum is 0, a new line is identified and separation is done. The sum of all rows in between two lines should be zero. STEP 4: For each line, the characters are to be extracted. This is done by finding the sum of all pixels value in a column. When sum is zero, a new character is identified and separation is done. STEP 5: Total 12 features value f1 to f12 are extracted for each character. STEP 6: The features value are matched with the trained features set to recognize the exact character. | Three fonts, namely ‘Arial’, ‘Berlin Sans’ and ‘Lucida Fax’ have been considered as training data set | Based on dataset used (Low) | Uses Feature extraction by finding number of corners  (Less accuracy for handwritten characters) | More datasets |
| 6 | Author: Raghuraj Singh and other  Year: June 2010  Paper: International Journal of Computer Science & Communication  Language: Devnagiri | Feature Extraction  • Mean Distance  • Histogram of projection based on spatial position of pixel  • Histogram of projection based on pixel value  ANN Approach for Classification: Training and recognition phase of the ANN has been performed using conventional back propagation algorithm with two hidden layers | The test set used in this experiment is of 77 characters of five different types of fonts. | Quite high | Less Dataset | Using good set of characters for classification. |
| 7 | Author: Chirag Patel and other  Year: October 2012  Paper: International Journal of Computer Applications  ISSN No. 0975 – 8887 | Image Pre-Processing  Using Tesseract | NIL | 100% for gray-scaled image  50% for coloured images | Less accuracy for coloured images | Convert coloured image to gray scale. |
| 8 | Author: Sang Sung Park and other  Year: December 2008  Paper: IJCSNS International Journal of Computer Science and Network Security | BP (Back Propagation) Algorithm | Self | Good | Less dataset less accuracy | More datasets |
| 9 | Author: Sameeksha Barve  Year: June 2012  ISSN: 2278 – 1323  Paper: International Journal of Advanced Research in Computer Engineering & Technology | The ANN is trained using the Back Propagation algorithm. In the proposed system, each typed English letter is represented by binary numbers that are used as input to a simple feature extraction system whose output, in addition to the input, are fed to an ANN. Afterwards, the Feed Forward Algorithm gives insight into the enter workings of a neural network followed by the Back Propagation Algorithm which compromises Training, Calculating Error, and Modifying Weights. | SELF | Pretty good | A poorly chosen set of features yields poor classification rates | Choosing more features enhances efficiency |
| 10 | Aauthor: Sukhpreet Singh  Year: June 2013  Paper: International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)  ISSN: 2278 – 1323 | Feature Extraction - This method defines each character by the presence or absence of key features, including height, width, density, loops, lines, stems and other character traits. Feature extraction is a perfect approach for OCR of magazines, laser print and high quality images. | SELF | Upto 95% for clear characters, 80% for old documents | Poor chosen key features can lead to poor efficiency | Choosing key correctly |
| 11 | Author: Dr.Mrs.V.V.Pati  Year: -February, 2015  Paper: International Journal of Engineering Research and General Science  ISSN 2091-2730 | TRAINING  a. Pre-processing – Processes the data so it is in a suitable form .  b. Feature extraction – Reduce the amount of data by extracting relevant information—Usually results in a vector of scalar values. (We also need to normalize the features for distance measurements)  c. Model Estimation – from the finite set of feature vectors, need to estimate a model (usually statistical) for each class of the training data. | SELF | Pretty good | A poorly chosen set of features will yield poor classification rates by any neural network | Choosing good set of features |
| 12 | Author: Abdelwadood Mesleh  Year: 2012  Paper: Contemporary Engineering Sciences | As Arabic characters are cursive, the dynamic-sized window may contain a full character of a portion of a character, to solve this problem, the proposed algorithm need to take a decision whether to accept or reject the character that is contained in the dynamic-sized window. If it is rejected, the width of the window is increased by some constant value (C) until a valid character is found (empirically, C is set to 5 pixels). It should be noted that the width of the window is initially set to a constant value=font size/10 and the search for candidate characters starts from right of each segmented and clipped line | SELF | 96.5% | Handling Arabic text with different orientation | Taking account of transformation angle using contours of text. |
| 13 | Author: C. VIGNESH and other  Year: APRIL 2015  Paper: International Journal of Emerging Technology in Computer Science & Electronics  ISSN: 0976-1353 | OPTICAL CHARACTER RECOGNITION FOR VISUALLY CHALLENGED PERSONS  The scanned image is taken as input. The image should be scanned with at least 300 dpi for better recognition result. The image is then preprocessed like grayscale conversion, binary conversion and so on. The preprocessed image is then given to the recognition engine for extracting the text from the image. The extracted text is further converted into Braille format and audio. | SELF | 94.64% for the 200 dpi, 96.58% for 300 dpi and 97.17% for 400 dpi | Less size image shows less accuracy | Converting the image to desired size |
| 14 | Improving OCR performance using character degradation models and boosting algorithm  By: Jianchang Mao ), K.M. Mohiuddin | Boosting algorithm:  Boosting is a machine learning ensemble meta-algorithm for primarily reducing bias, and also variance in supervised learning, and a family of machine learning algorithms that convert weak learners to strong ones.  The standard backpropagation algorithm is used to train all the individual networks. | Alphabets of the NIST. National Institute of Standards and Technology Special Database 3 | There is seen a improve in performance rate by 93.86%. | Time taken for analysing increased over 3 times. | Boostingalgorithm can be parallelized using Map reduce, so that it can be more optimized. |
| 15 | Author: Mande Shen and other  Year: January 2016  Paper: 2015 12th International Conference on Fuzzy Systems and Knowledge Discovery  ISBN: 978-1-4673-7682-2 | Improving OCR Performance with Background  Image Elimination.  This method is based on the difference of the values of the R, G, B colours in background image pixels. In this method, brightness distortion and chromaticity are used to enhance the  contrast in each channel image. | The dataset consists of 1160 images with various resolutions, font sizes, illumination contrast, background complexity and noise levels. | Pretty Good for coloured background | Algorithm assumes the background images are colourful, does not work for black and white background. | Binarizing image in pre-processing steps. |
| 16 | Author: Shalini Sonth and other  Year: February 2018  Paper: 2017 International Conference on Electrical, Electronics, Communication, Computer, and Optimization Techniques  ISBN: 978-1-5386-2361-9 | OCR Based Facilitator for the Visually Challenged.  Using Raspberry Pi and using extracted text to convert to speech.  Pre-Processing  Text extraction using Tesseract OCR. | NIL | Good for normal text image | OCR fails to carry out the image to text conversion with a good accuracy when the text is italicized.  If the image is composed of more than colours, OCR fails to perform image to text conversion with good accuracy. | More variety of datasets or adding different algorithm for error detection. |
| 17 | Author: Paul Thompson and other  Year: February 2016  Paper: 2015 Digital Heritage  ISBN: 978-1-5090-0048-7 | Customised OCR Correction for Historical Medical Text.  Error correction:  Spell checker evaluation, using modifying Hunspell (<http://hunspell.sourceforge.net/>),  Re-ranked correction suggestions based on their frequencies of occurrence in the BMJ archive.  Method helps to filter out suggestions that are highly unlikely in the context of medical text. | BMJ archive of two decades. | Worst -> 16% error removal, | Only useful for detection of same decade medical data, because of frequency of Hunspell’s dictionary. | NIL |
| 18 | Author: Kyamelia Royand other  Year: April 2018  Paper: 2017 4th International Conference on Opto-Electronics and Applied Optics  ISBN: 978-1-5386-1119-7 | An efficient OCR Based Technique for Barcode Reading and Editing.  Loading the image  Binarization (Skew detection & Correction method)  Filtering (noise removal)  Contrast Enhancement (histogram equalisation technique)  For decoding the barcode, the intensity on the columns of the image is computed and the value is returned in an array. The algorithm then computes the number of black and white bars and stores the number of bars into a variable. Subsequently, the number of bits is computed by the third function and the bit information aggregated is send back. The minimum and maximum value stored in the array is generally taken and compared. | NIL | Very Good | NIL |  |

**Base Paper:** A Font style classification system for English OCR

**Author:** Bharath V & N. Shobha Rani

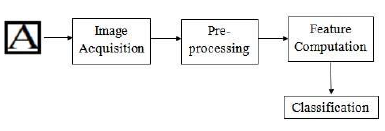
**Year Published:** March 2018

Various steps which are performed are as follows:

* **Input:** also, can be told as collection of data i.e., different type of images and different type of formats.
* **Image processing:** once the images are collected the images are processed like noise removal, skew correction, brightness, contrast etc.
* **Document and layout analysis:** once the quality of the image is been enhanced it is then analysed for the line spacing, word spacing etc.
* **Recognition:** after the analysis is done the image is subjected for recognition like font style, font size etc.
* **Verification and user interaction:** once the recognition is done the system provides basic information like character coordinates, word and character recognition hypothesis etc.
* **Output:** after all the process is done the output is obtained.

**Proposed Methodology:** The proposed system for font style recognition and classification system mainly involves three main stages. In the stage 1, an input image is acquired for processing and proceeded for pre-processing and further directed for feature computation in stage two. Finally, the features computed are classified through an SVM classifier.

**PRE-****PROCESSING:** Each image is initially resized to a fixed dimension of around 35x35 so as to obtain consistent number of features for all inputs and maintain the robustness of system with respect to various changes that occurs in inputs. The rescaled images are transformed to binary images through Otsu’s thresholding approach. The pre-processed images are directed towards the feature computation stage subsequently.



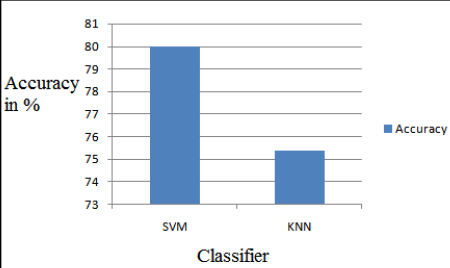
**FEATURE COMPUTATION:** Feature computation involves the process of extracting unique features from pre-processed images so that it can be employed for classification of fonts. In the proposed method a hybrid combination of features by extracting a set of distance profile features from pre-processed images. The combination of features computed from images include left distance profile, right distance profile and diagonal distance profile features.

**Left distance profile features:** Left distance profile feature is the distance between first column of the character image to first outer border pixel. The distance is computed as the sum of the pixels that are lying in between the first column and first outer border pixel of character image.

**Right distance profile features:** Right distance profile features are the distance between the last outer border pixel of a character image to the last column of the character image. The distance is computed as the sum of the pixels between the last outer border pixel and the last column of the character image.

**Diagonal distance profile features:** Diagonal distance profile features represents the diagonal distance from first pixel at position Ci(1,1) (top left corner of image) to the outer border pixel of character in an image Ci by computing the sum of pixels in diagonal directions.

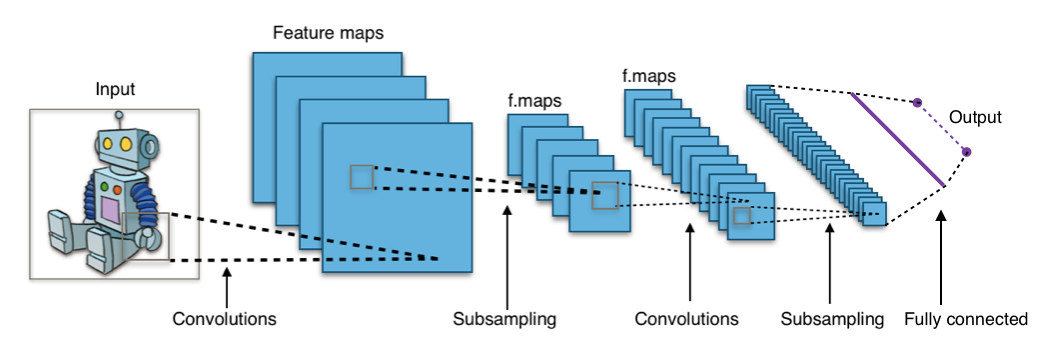
**Accuracy Comparison:**

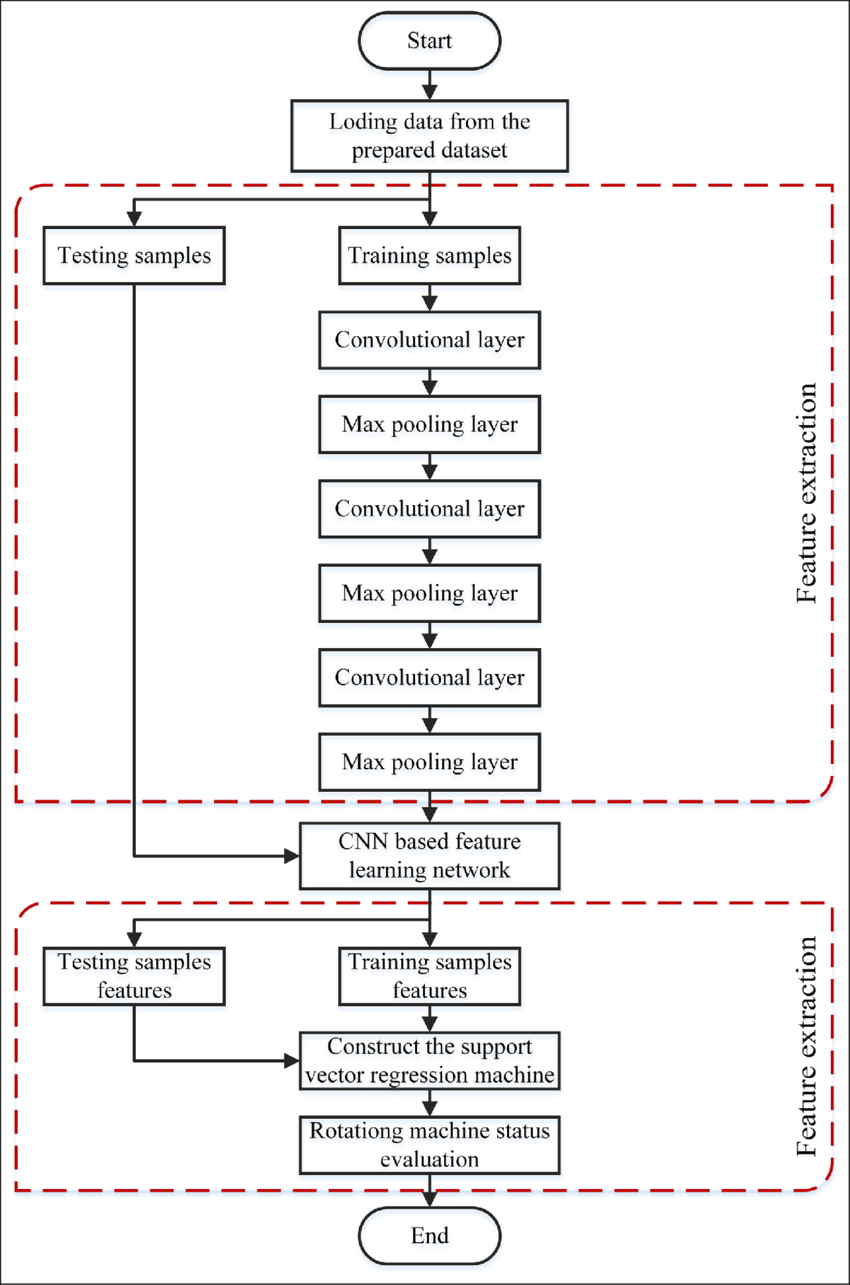
**The accuracy of the system was found to be around 80%**

**Algorithm Description**

The algorithm used in our system is Convolution Neural Network (CNN). It's a deep, feed-forward artificial neural network. Remember that feed-forward neural networks are also called multi-layer perceptron’s (MLPs), which are the quintessential deep learning models. The models are called "feed-forward" because information flows right through the model. There are no feedback connections in which outputs of the model are fed back into itself.

CNNs specifically are inspired by the biological visual cortex. The cortex has small regions of cells that are sensitive to the specific areas of the visual field. This idea was expanded by a captivating experiment done by Hubel and Wiesel in 1962 (if you want to know more, here's a video). In this experiment, the researchers showed that some individual neurons in the brain activated or fired only in the presence of edges of a particular orientation like vertical or horizontal edges. For example, some neurons fired when exposed to vertical sides and some when shown a horizontal edge. Hubel and Wiesel found that all of these neurons were well ordered in a columnar fashion and that together they were able to produce visual perception. This idea of specialized components inside of a system having specific tasks is one that machines use as well and one that you can also find back in CNNs.





**Proposed Architecture**

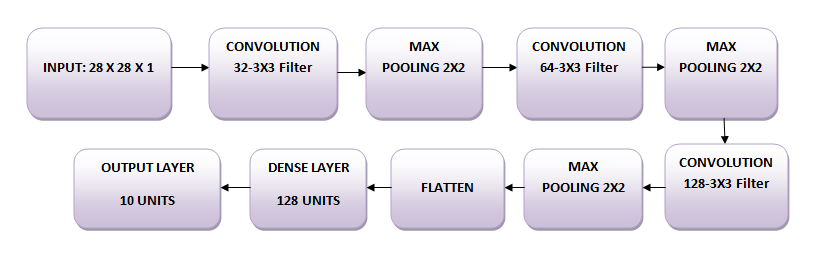
The architecture of this system comprises of 3 steps:

1. Image pre-processing
2. Conversion to csv file and splitting
3. Training and testing using CNN

**Image pre-processing:** This step involves the pre-processing of all image files using opencv. This step involves the noise removal, filtering, resizing and feature extraction of image files. Then the images are stored in different folder.

**Conversion to csv file and splitting:** This step involves making of csv file from all image files by converting them into 1-d matrix using OpenCV module in python. Then the produced dataframes are shuffled for getting better accuracy and then dataframes are divided into 2 sets of training set and testing set with the ratio of 70:30 and two csv files are generated.

**Training and testing using CNN:** The testing images are trained using batch size of 64 with 20 epochs.



**Flow-chart:**

Load Training + Testing Dataset

Test Data

Train Data

**Code & Results**

1. Code:

opencv.py





pre\_processing.py





generating\_csv.py

****

****

****

split.py

****

****

cnn.py

import keras

from keras.models import Sequential

from keras.layers.core import Dense, Dropout, Activation, Flatten

from keras.layers.convolutional import Convolution2D, MaxPooling2D

from sklearn.model\_selection import train\_test\_split

from keras.models import Input,Model

from keras.layers import Conv2D

from keras.layers.normalization import BatchNormalization

from keras.layers.advanced\_activations import LeakyReLU

import pandas as pd

import numpy as np

#read training data

data = pd.read\_csv('data\_final\_train.csv')

data.head()

#reshaping into 28X28 array

data.iloc[3,1:].values.reshape(28,28).astype('uint8')

#Storing Pixel array in form length width and channel in df\_x

df\_x = data.iloc[:,1:].values.reshape(len(data),28,28,1)

#Storing the labels in y

y = data.iloc[:,0].values

#converting labels to categorical data

df\_y = keras.utils.to\_categorical(y)

#normalizing image values (0 to 1)

df\_x = np.array(df\_x)

df\_y = np.array(df\_y)

df\_x = df\_x.astype('float32')

df\_x = df\_x / 255.

#spliting training data to 2 parts (training and validation to prevent over-fitting)

train\_X,valid\_X,train\_label,valid\_label = train\_test\_split(df\_x,df\_y,test\_size=0.2,random\_state=13)

batch\_size = 64

epochs = 20

num\_classes = 62

#initializing sequential model

model = Sequential()

#adding convolution layer (filter,shape,activation,input\_shape,padding)

model.add(Conv2D(32, kernel\_size=(3, 3),activation='linear',input\_shape=(28,28,1),padding='same'))

#pooling operation is to reduce the size of the images

model.add(MaxPooling2D((2, 2),padding='same'))

#converting image pixels to a 1-D single vector

model.add(Flatten())

#connecting layers with number of nodes in hidden layer

model.add(Dense(128, activation='linear'))

#initialize output layer

model.add(Dense(num\_classes, activation='softmax'))

#compiling the model

model.compile(loss=keras.losses.categorical\_crossentropy, optimizer=keras.optimizers.Adam(),metrics=['accuracy'])

#training the model

train\_model = model.fit(train\_X, train\_label, batch\_size=batch\_size,epochs=epochs,validation\_data=(valid\_X, valid\_label))

#read test data

data\_test = pd.read\_csv('data\_final\_test.csv')

data\_test.head()

#reshaping into 28X28 array

data\_test.iloc[3,1:].values.reshape(28,28).astype('uint8')

#Storing Pixel array in form length width and channel in df\_x

df\_x\_test = data\_test.iloc[:,1:].values.reshape(len(data\_test),28,28,1)

#Storing the labels in y

y\_test = data\_test.iloc[:,0].values

#normalizing pixel values (0 to 1)

df\_y\_test = keras.utils.to\_categorical(y\_test)

df\_x\_test = np.array(df\_x\_test)

df\_y\_test = np.array(df\_y\_test)

df\_x\_test = df\_x\_test.astype('float32')

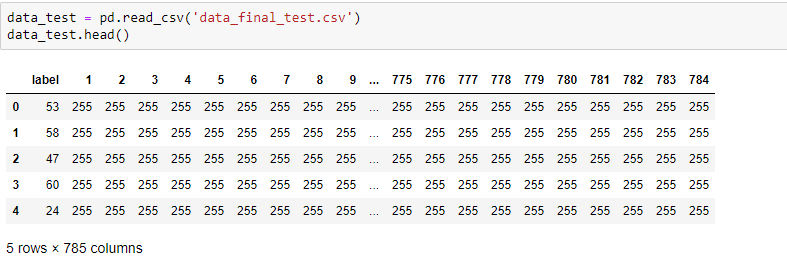
df\_x\_test = df\_x\_test / 255.

#predicting the test data

test\_eval = model.evaluate(df\_x\_test, df\_y\_test, verbose=1)

print('Test loss:', test\_eval[0])

print('Test accuracy:', test\_eval[1])



**2) Results**

Training dataset by splitting it in 2 parts (training and validation) to remove over-fitting.

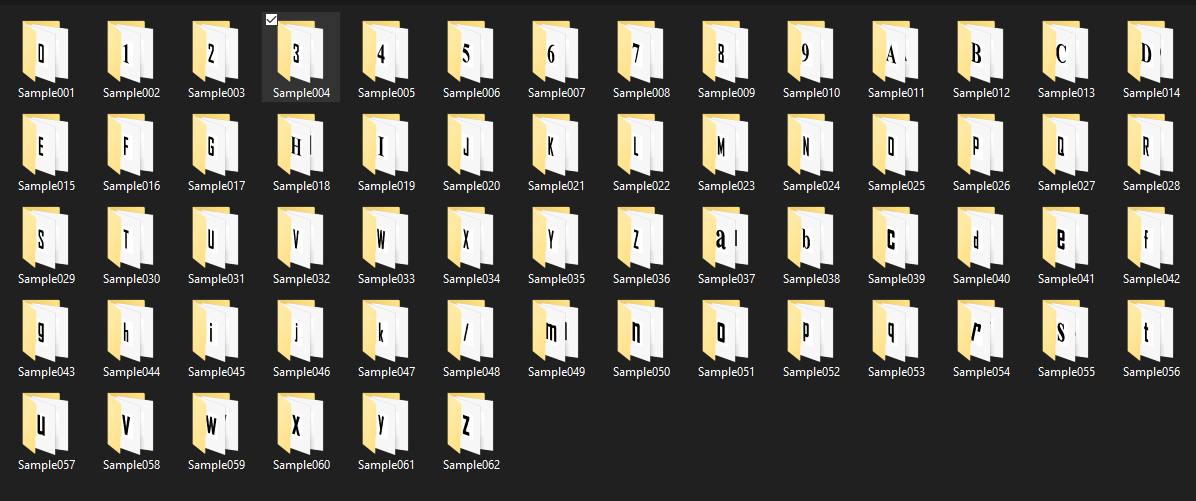
|  |  |  |
| --- | --- | --- |
| Epochs | Time | Accuracy |
| 1 | 60s | 0.667 |
| 2 | 63s | 0.8172 |
| 3 | 56s | 0.8427 |
| 4 | 55s | 0.8632 |
| 5 | 56s | 0.8786 |
| 6 | 56s | 0.8923 |
| 7 | 56s | 0.9008 |
| 8 | 57s | 0.9089 |
| 9 | 57s | 0.9170 |
| 10 | 60s | 0.9210 |
| 11 | 62s | 0.9256 |
| 12 | 65s | 0.9299 |
| 13 | 70s | 0.9350 |
| 14 | 65s | 0.9368 |
| 15 | 73s | 0.9406 |
| 16 | 70s | 0.9429 |
| 17 | 61s | 0.9440 |
| 18 | 60s | 0.9464 |
| 19 | 60s | 0.9487 |
| 20 | 61s | 0.9504 |

Accuracy on the testing set:

|  |  |  |
| --- | --- | --- |
| Dataset | Accuracy (Proposed) | Accuracy (Existing) |
| 18898 | 87.18% | 80% |

**3) Screenshots**

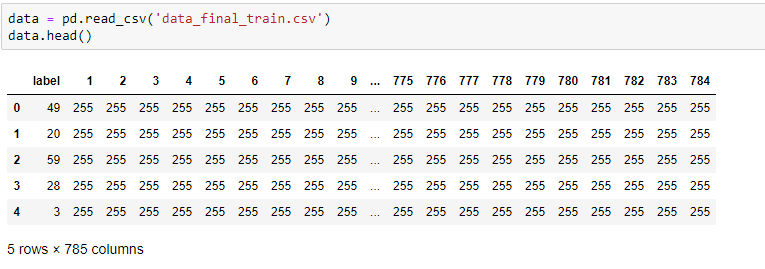
Initial images:



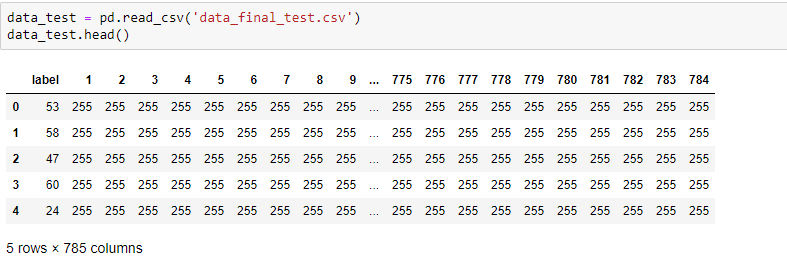
Processed images:



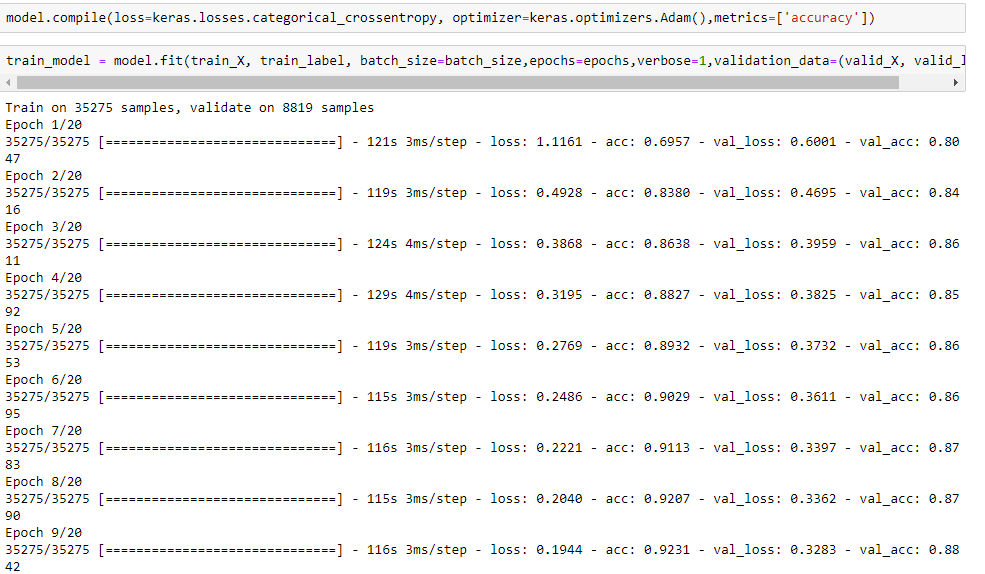
Training dataset head:

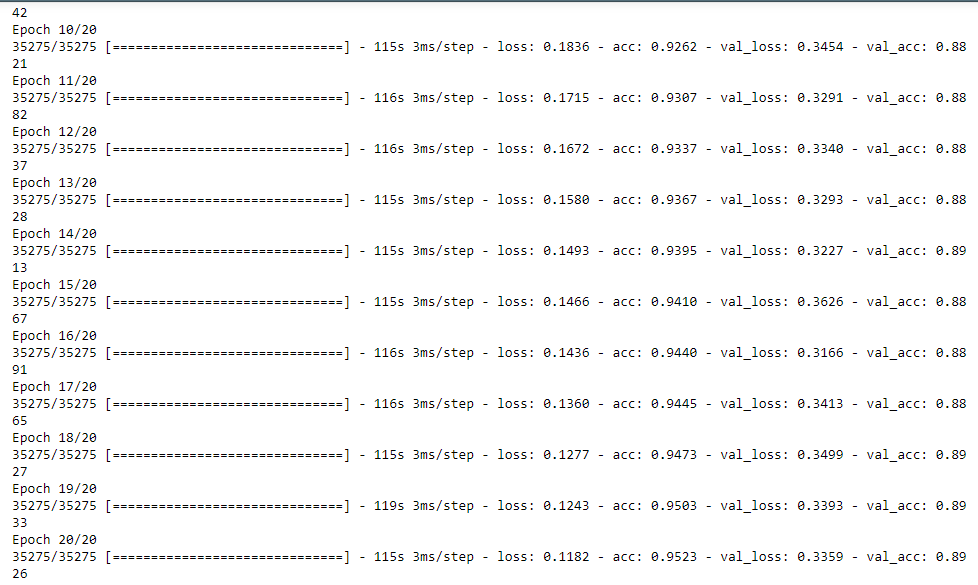


Testing dataset head:

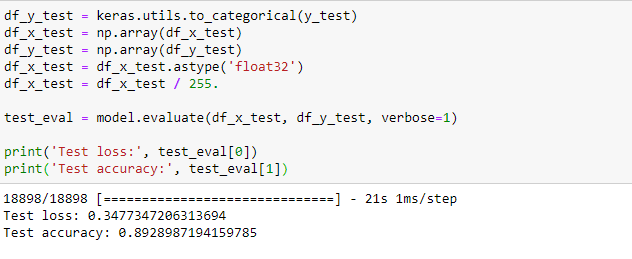


Training using CNN:





Accuracy of testing dataset:



**Conclusion**

We have successfully developed the system to train the image for the optical character recognition with a good accuracy of about 89.28%, which is far better than the one we saw in our base paper which had an accuracy of around 80%. So, we can claim that using neural network instead of classification algorithm helps to improve accuracy of testing dataset.

**Future Work**

The optical character recognition system can be enhanced by developing an algorithm which is faster and more accurate and using more precision in the pre-processing steps. Another thing which to be done is making of the user-friendly software and integrate it with mobile applications to make this system useful and it can also be combined with text-to-voice system so that the recognised text can be converted into voice and can be used by people which are not able to see.

**References**

[1] T. E. de Campos, B. R. Babu, and M. Varma. Character recognition in natural images. In

Proceedings of the International Conference on Computer Vision Theory and Applications, Lisbon,

Portugal, February 2009.

[2] Muhammad Fraz, M Saquib Sarfraz, Eran A Edirisinghe, and UK Loughborough. Exploiting

colour information for better scene text recognition.

[3] Deepak Kumar and AG Ramakrishnan. Recognition of kannada characters extracted from

scene images. In Proceeding of the workshop on Document Analysis and Recognition, pages

15–21. ACM, 2012.

[4] Lukas Neumann and Jiri Matas. A method for text localization and recognition in realworld

images. In Computer Vision–ACCV 2010, pages 770–783. Springer, 2011.

[5] Andrew J Newell and Lewis D Griffin. Natural image character recognition using oriented

basic image features. In Digital Image Computing Techniques and Applications (DICTA), 2011

International Conference on, pages 191–196. IEEE, 2011.

[6] Karthik Sheshadri and Santosh Kumar Divvala. Exemplar driven character recognition in

the wild. In BMVC, pages 1–10, 2012.

[7] Cun-Zhao SHI, Chun-Heng WANG, Bai-Hua XIAO, Yang ZHANG, and Song GAO. Multiscale

graph-matching based kernel for character recognition from natural scenes. Acta Automatica

Sinica, 40(4):751–756, 2014.

[8] Ziming Zhang, Paul Sturgess, Sunando Sengupta, Nigel Crook, and Philip Torr. Efficient

discriminative learning of parametric nearest neighbor classifiers. 2012.