Text recognition Using Convolutional Neural Network.

[1]Prof.A.H.Kadam,[2]Rutuja Medhekar,[3]Anushree Chopde,[4]Sijin Saji, [5]Tejesh Shelake

[1]Professor,[2] Student,[3]Student ,[4]Student, [5]Student
[1]Department of Information Technology,
[1]Rajarshi Shahu College Of Engineering, Pune, India.

Abstract
Text recognition is gaining a huge demand in the branch of computer vision. OCR is the transformation of images of text into machine encoded text. A simple API to an OCR library might provide a function which takes as input an image and outputs a string. We are going to implement a better and accurate approach to perceive and foresee various images having different kinds of fonts, digits,etc.. A class of multilayer sustain forward system called Convolutional network is taken into consideration. A Convolutional network has a benefit over other Artificial Neural networks in extracting and utilizing the features data with higher degree of accuracy and unvarying to translation,scaling and other distortions.We have multiple OCR's available in market but our aim is to build our own OCR and try to have the accuracy of the obtained results approximately equivalent to the Open source OCR engine i.eTesseract. The primary aim of this dataset is to classify the digits and text for particular font style . We have a total of 50,000 images for training and testing.We are going to implement this network using tensorflow deep learning inbuilt python library and deploy the results using android mobile app.

IndexTerms–Convolutional Neural Network, python library, OCR Engine, multilayer system.

I. INTRODUCTION

Research carried out in the field of Optical Character Recognition (OCR) was restricted to document images acquired with the desktop scanners in the past few decades. The usage of such systems is restricted as they are not portable because of very large size of the scanners and the need of a computing system. Moreover, the shot speed of a scanner is slower than that of a digital camera. Recently, with the advancement of processing speed and internal memory of hand-held mobile devices such as smart phones, iPhones, etc. having built-in digital cameras, a new trend of research has emerged into picture. Researchers have dared to think of running OCR applications on such devices for having real time results.
Convolutional neural networks (convnets) are used in situations where data can be expressed as a "map" where the proximity between two data points indicates how related they are. An image is such a map, which is why you so often hear of convnets in the context of image analysis. If you take an image and randomly rearrange all of its pixels, it is no longer recognizable. The relative position of the pixels to one another, that is, the order, is significant.

Convnets are commonly used to categorize things in images, so that's the context in which we'll discuss them**.**
A convnet takes an image expressed as an array of numbers, applies a series of operations to that array and, at the end, returns the probability that an object in the image belongs to a particular class of objects. For instance, a convnet can let you know the probability that a photo you took contains a building or a horse or what have you. It might be used to distinguish between very similar instances of something. For example, you could use a convnet to go through a collection of images of skin lesions and classify the lesions as benign or malignant.
Convnets contain one or more of each of the following layers:

- convolution layer
- ReLU (rectified linear units) layer
- pooling layer
- fully connected layer
- loss layer (during the training process)

There have been many applications developed that uses OCR engine for scanning the documents and this applications are using simple machine learning algorithm i.e. multi-stage processes. In this, first the image may be divided into smaller regions that contain the individual characters, second the individual characters are recognized, and finally the result is pieced back together. A difficulty with this approach is to obtain a good division of the original image. But in our project we are developing our own OCR engine based on Convolutional neural network which increases the accuracy of the characters and digits efficiently rather than the available engines in the market. The major advantage of our OCR engine is that it will be an open source android app which will help the end-users to effectively use the app for scanning the documents.

II. STRUCTURE OF OUR SYSTEM

The below given is our proposed system architecture which consists client that maybe any android application user which gets connected to the server using http. Server has all the functionality and operationalities for processing the client's request and serves the client by sending the result of scanned images.

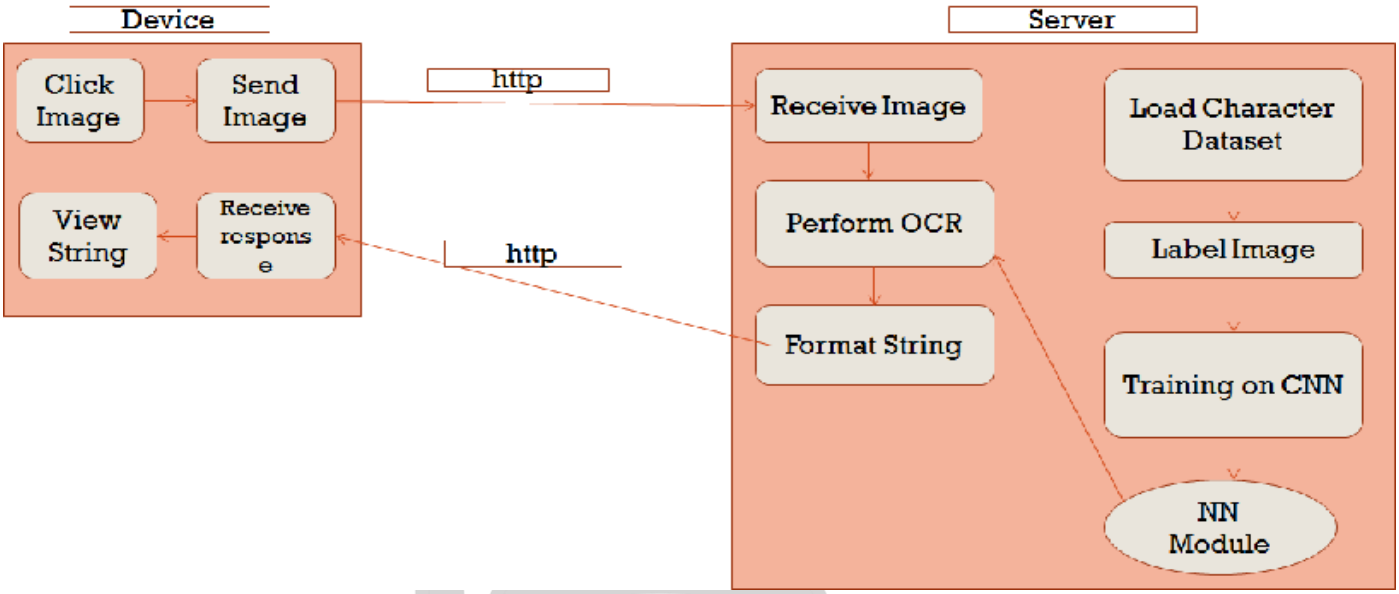

Figure 1:Architecture of our system

III. WORKING OF OUR SYSTEM

A complete working of theconvolutional network is presented that transforms the input volume into a sequence of character predictions. These character predictions can later be transformed into a string as shown in thediagram below:

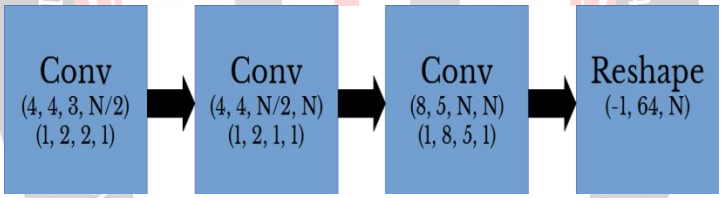

Figure 2 :Working of Convolutional layers

Where N is the number of possible characters. Let us consider,there are 63 possible characters for uppercase and lowercase characters, digits. The parenthesized values in the given convolutional layers are the actually filter sizes and stride values from top to bottom respectively. The values that are present in the reshape layer are the reshaped dimension.

The input volume is a rectangular RGB image. In this first height and width of this volume are reduced across the convolutional layers using striding. The 3rd dimension of the volume is increased from 3 channels (RGB) to 1 channel for every possible character. Thus, the volume is further transformed from an RGB image into the sequence of vectors. Now, applying *argmax* across the generated channel dimension will give us a sequence of 1 hot encoded vectors which can be transformed into a string.

3.1Generating data

In order to fease out the process of training this network, we generate a dataset using the Python Imaging Library (PIL). In this, random strings which consist of alphanumeric characters are generated. By using PIL, images are generated for each random string. In addition to this, a CSV file is also generated which contains the particular file name and the random string assossiated with it.

3.2Training data

- In order to train the network, the generated CSV file is parsed and the images are loaded into memory. Each target value required for the training data is a sequence of 1-hot vectors. Thus the target matrix we required is a 3D matrix with the three dimensions corresponding to sample, character, and 1-hot encoding respectively.
- In the next step, the neural network is constructed using the artificial neural network classifier (ANNC) class from TFANN. (Tensor Flow Artificial Neural Network).

- For this we have the Softmax cross-entropy that is used as the loss function which is performed over the 3rd dimension of the output.
- With the help of ANNC class the process of fitting the network and performing the predictions becomes simple. The generated prediction is next split up using array_split from numpy to prevent any memory errors.

IV. METHODOLOGY

The proposed network consists of seven different heterogeneous layers. Each layer contains feature maps which are the results of convolution, subsampling, or neuron unit operations. Applying and combining these automatically learnt operations ensure the extraction of robust features, leading to the automatic recognition of characters in natural images.

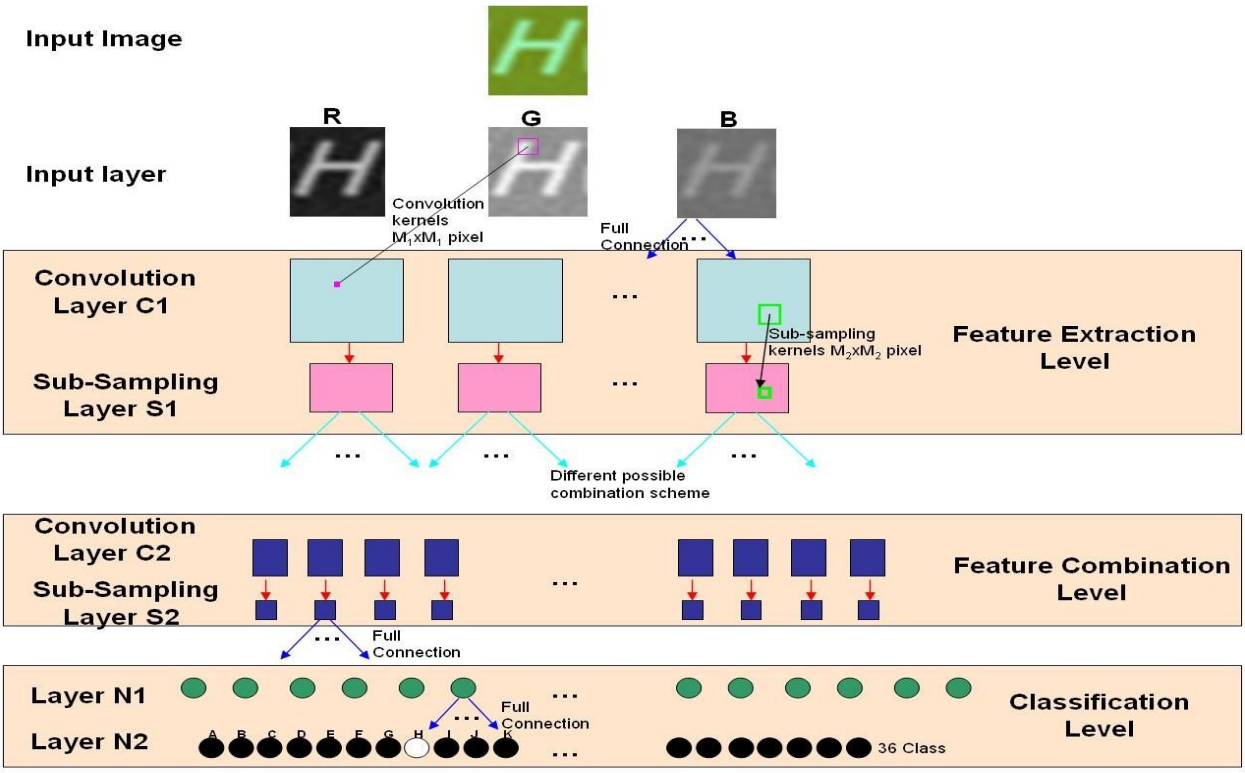

Figure 3: Architecture of the Network

The first layer is the input layer E; it consists of NE = 3 input maps, each of them corresponding to one colour channel of the image, depending on the colour space (RGB, YUV, CMY, etc.). Their pixel values are normalized to the range [-1, 1].
The RGB colour space has been chosen in our experiments. We can distinguish three main levels:

I. **LEVEL 1**: Feature extraction level, relying on the C1 and S1 layers. The layer C1 extracts some specific features (oriented edges, corners, end points) directly from the three colour channels. In fact, this layer is composed of NC1 maps. Each unit in each map is connected to a M1 × M1 neighbourhood (biological local receptive field) in each of the colour channels of the input layer. Furthermore, the trainable weights (convolutional mask) forming the receptive field, are forced to be equal for the entire map (weight sharing).
A bias that can be trained is added to the results of each convolutional mask. Thus, each map can be considered as a feature map that has a learnt fixed feature extractor that corresponds to a pure convolution with a trainable mask, applied over the channels in the input layer.
Multiple maps lead to the extraction of multiple features. Therefore, each map of the third layer S1 results from local averaging and sub-sampling operations on a corresponding map in the previous layer C1. So, the layer S1 is composed of NS1 = NC1 feature maps.
We use this sub-sampling layer to reduce by two the spatial resolution which reduces the sensitivity to shifts, distortions and variations in scale and rotation.

II. **LEVEL 2**: Feature combination level, relying on the C2 and S2 layers. Layer C2 allows extracting more complex information; outputs of different feature maps of layer S1 are fused in order to combine different features. As in the first level, in this second level also, we consider a sub-sampling layer S2, where each map results from local averaging and sub-sampling operations applied on a corresponding map in the previous layer C2.
In spite of, this progressive reduction of spatial resolution that is compensated by a progressive increase of the richness of the representation, which actually corresponds to the number of feature maps and enables a large degree of invariance to geometric transformations of the input.

III. **LEVEL 3**: Classification level, relying on the N1 and N2 layers. Each of them is a fully connected layer and contains classical neural units. The choice of the number of units in N1 is empirical; whereas the number of units in N2 depends on the number of classes to be recognized. If we consider the alphanumerical patterns, we will get 62 classes (26 lower case characters, 26 upper case characters and 10 digits).
In order to reduce the number of output classes and consequently the number of neural weights to be learnt in between layers N1 and N2, we propose to use only 36 classes, where corresponding upper case and lower case characters are fused. This is made possible thanks to the strong generalization abilities of the proposed network

.

4.1 Training the network

We choose to normalize the size of all the images to 48 lines by 48 columns in RGB color space.

As mentioned before, we use 34845 color character scene images, Nt = 30000 in the training set andNv = 4845 in the validation set. In fact, to avoid over fitting the training data and to increase the generalization ability of the system, a part of the whole training set is kept as validation set. This validation set is used to evaluate the network performance through iterations, as explained later on.

We choose to build NC1 = NS1 = 6 maps in layers C1, and S1; NC2 = NS2 = 16 maps in layers C2, and S2; 120 units in N1; and NClass = 36 units in layer N2. The combination scheme used is the following (figure 3): The first six C2 feature maps take inputs from every contiguous subset of three feature maps in S1. The next six take input from every contiguous subset of four feature maps in S1. The next three take input from some discontinuous subsets of four feature maps in S1. Finally, the last one takes input from all S1 feature maps.

The convolution window size M1×M1=M2×M2=5×5 for both convolution layers C1 and C2. The sub-sampling factor is two in each direction for both sub-sampling layers S1 and S2. We use linear activation functions in C1 and C2 and sigmoid activationfunctions in S1, S2, N1 and N2.

The different parameters governing the proposed architecture, i.e., the number of layers, the number of maps, as well as the size of the receptive fields, have been experimentally chosen. The training phase was performed using the classical back-propagation algorithm with momentum modified for being used in convolutional networks as described and it consists of the following steps:

1.Construct the desired output vector {Dh}h=1..NClass : for a given character image, belonging to class h, this desired output vector is constructed by setting its hth element to 1, and setting the remaining elements to -1.

2. Compute the actual output vector {Oh}h=1..NClass : the character image is presented to the network as input, the last layer output vector represents the actual output vector.

3. Update weights: the weights are updated by back propagation of the error between the actual output vector and the desired output vector.

4. Repeat from step 1 to the step 3 for the Nt character images of the training set.

5. Compute the MSE (Mean Square Error) over the validation set: for every character images of the validation set, repeat step 1 and 2, and then compute the MSE between the actual output vectors $\{O_{h,k}\}_{h=1..N_{Class},k=1..N_v}$ and the desired output vectors $\{D_{h,k}\}_{h=1..N_{Class},k=1..N_v}$ as follow:

$$MSE = \frac{1}{N_v \times N_{Class}} \sum_{k=1}^{N_v} \sum_{h=1}^{N_{Class}} (O_{h,k} - D_{h,k})^2$$

6. Save the weights values if MSE is decreasing.

7. Repeat steps 1 until 6, until the desired number of iterations is reached. After training, the system is ready to recognize automatically and rapidly color character images, without the need of any binarization preprocessing. In fact, this system is actually able to produce directly an output vector {Oh}h=1..NClass for a given color input character image. The index which corresponds to the highest component of this vector is then considered as the recognized class. After 50 training iterations, the proposed network achieves an average recognition rate of 90.05% on the entire training set.

IV. RESULTS

Training and cross-validation results are shown below in Figure on the left and right respectively. The graphs are shown separately as the plots nearly coincide.

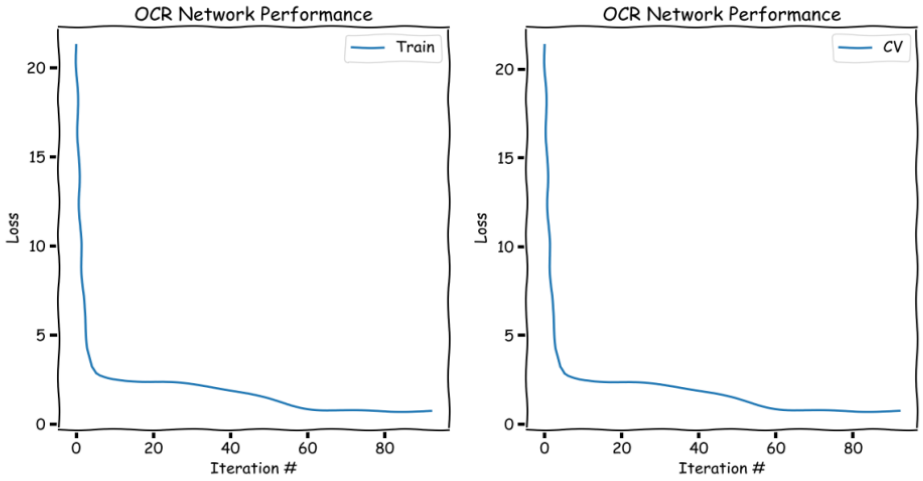

Figure 4: Network Training Performance

Figure below shows the performance of the network for several images from the dataset.

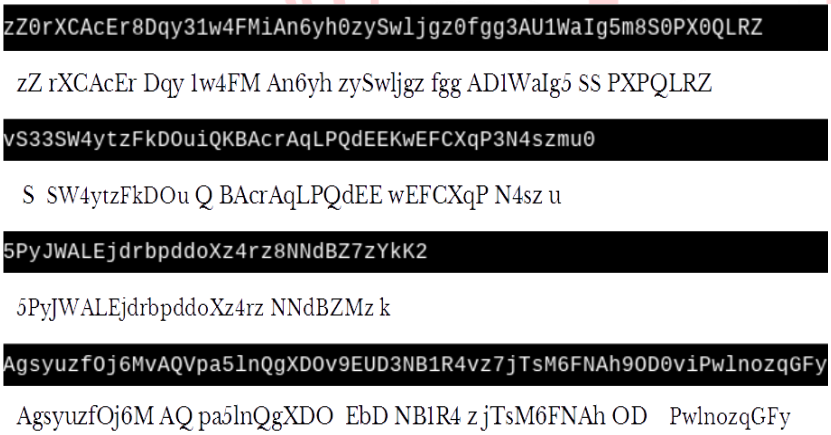

Figure5 : Sample OCR Results

The text part below each image is the predicted text that is generated by the network. In order to check working of the code we performed running it on a laptop which had high integrated graphics and so the amount of data and size of the network was constrained for performance reasons. Further improvements can be likely be made to increase the performance with a larger dataset and network.

V. CONCLUSION

A complete OCR system has been presented in this paper.
Performance of the current system is observed by taking into consideration the variations in number of iterations and the variations in the number of characters. The proposed system produce 60 % recognition rate for fonts by considering up to 64 characters at a time. From this we can use this OCR System for recreating the historical documents again which will massively reduce the time required for typing the documents in office, college, etc. As this recent technology is under research in any companies and institutes.
Recognition is often followed by a post-processing stage. We hope and foresee that if post-processing is done, the accuracy will be even higher and then it could be directly implemented on mobile devices. Implementing the presented system with post-processing on mobile devices is also taken as part of our future work.
.

REFERENCES

[1] Honey Mehta ,Sanjay Singla andAartiMahajan, "OpticalCharacter Recognition System for Roman script and English Languages Using ANN ", in International Conference on Research Advance In Integrated Navigation System (RAINS-2016).

[2] Youn Jing, BaharYouseffi and MitraMirhassani, "An Efficient FPGA Implementation of Optical Character Recognition for License Plate Recognition", in 2017 IEEE 30th Canadian Conference on Electrical and Computer Engineering (CCECE)

[3]KiranBhadwaik, KhalidMahsud and AsidRaza, "Autonomous Decentralized Systems, towards applying OCR and semantic web to achieve optimal learning experience.", in 2017 IEEE 13th International Symposium

[4]J. Mercy Geraldine andAnu Disney, "A Novel Approach for Mining Relevant Frequent Patterns in an Incremental Database", in IOSR Journal of Computer Engineering (IOSR-JCE) e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 17, Issue 2, Ver. II (Mar – Apr. 2015), PP 22-26 .

[5]Md. Rabiul Islam, ChayanMondall and Md. KawsarAzam, "Text Detection and Recognition Using Enhanced MSER Detection and a Novel OCR Technique", in 2016 5th International Conference on Informatics, Electronics and Vision (ICIEV)

[6]M UsmanRaza, et al., "Text Extraction Using Artificial Neural Networks", in Networked Computing and Advanced Information Management (NCM) 7th International Conference,Gyeongju, North Gyeongsang, 2011, pp. 134 - 137.

[7]Fonseca, J.M., et al., "Optical Character Recognition Using Automatically Generated Fuzzy Classifiers", in Eighth International Conference on Fuzzy Systems and Knowledge Discovery, Shanghai, 2011, pp. 448 – 452.

