Digit Recognizer

Rishika Machina
Computer Engineering
San Jose State University
San Jose, CA
rishika.machina@sjsu.edu

Tanmay Raoji Shinde
Computer Engineering
San Jose State University
San Jose, CA
tanmayraoji.shinde@sjsu.edu

Sampath Lakkaraju
Computer Engineering
San Jose State University
San Jose, CA
sampath.lakkaraju@sjsu.edu

Hansraj Pabbati
Computer Engineering
San Jose State University
San Jose, CA
hansraj.pabbati@sjsu.edu

***Abstract*— Handwritten digit recognition is one of the important research topics in computer vision and pattern recognition. An effective approach has been developed for handwritten digit recognizer based on multi feature extraction and deep analysis. Normalization of images has been done which is followed by preprocessing of data and keep relevant features. Handwritten image recognition is different from traditional image semantics recognition for which specific feature definitions, including structure features are required.**

Moreover, we fuse multiple features into deep neural networks for recognition of images. Testing was done on self-extracted image database combined with MNIST (Modified National Institute of Standards and Technology database) handwritten digit images to show that performance of our algorithm is remarkable and demonstrated superiority over several existing algorithms.

Keywords—Handwritten Digit Recognition, multi-feature classification, deep neural networks.

I. INTRODUCTION

Handwritten digit recognition plays an important role in pattern recognition and Optical Character Recognition (OCR) [1]. It has a wide range of practical applications when it comes to real life applications, such as zip code recognition in postal mail sorting, form processing and handwritten digits recognition on bank check etc. As a part of Offline recognition system, the handwritten image can be scanned offline from the document by optical character recognition (O.C.R) whereas in case of online recognition text can be taken form special digitizer or Personal Digital Assistant.

Over the past decades, Many machine learning methods have been employed for effective handwritten digit recognition, such as Linear and Non-Linear classifier, support vector machines(SVMs), Convolution Neural Networks etc.

Digit recognition is the process of taking pixel data of an image and converting it into readable format. There are certain digits like: 3, 8 and 6 which have some similarities. The challenge in this task is to differentiate them using suitable algorithms. There are several different ways to recognize any pattern depending on dataset chosen. We considered MNIST Kaggle dataset to train and test the artificial neural networks model. The considered dataset consists of 42000 rows of training data and 28000 rows of testing data in which handwritten digits ranging from 0 to 9 are present.

II. WORKING PRINCIPLE

The steps involved in this project are divided into six phases which are image acquisition, pre-processing, segmentation, feature extraction, classification and post processing respectively. The block diagram of the basic digit recognition is shown in Fig [1].

A. Image Acquisition

The most common devices from which we can take input images are tablet, digitizer or Personal Digital Assistant. Input images can also be taken from devices such as scanners, photographs etc.

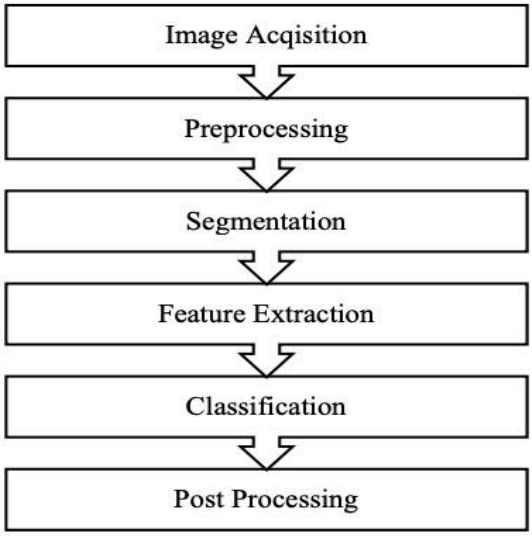

Figure 1: Project pipeline

B. Preprocessing

Pre-processing is one of the crucial phases as absence of this step might result in low accuracy for any algorithm. The main objective of this step is to normalize the data and reduce distortions due to noise. Pre-processing includes five common steps, namely, size normalization and centering, interpolating missing points, smoothing, slant correction and resampling of points. Depending on the data used, the steps used in pre-processing might change.

C. Segmentation

Segmentation is separation of an image. Document is processed in hierarchical way. At first, lines are segmented and then from each row pixel data is extracted.

D. Feature Extraction

The aim for the feature extraction is to extract the pattern from the image which is most pertinent for classification. These features are used to train the model.

E. Classification

Input image is presented to the system and its features are extracted. Features and Labels are used to train the model. Classifiers compare the input feature with stored pattern and find out the best matching class for input given from the test data.

F. Post Processing

Post-processing refers to the procedure of correcting misclassified results by applying linguistic knowledge. Postprocessing is processing of the output from shape recognition.

III. SOLUTION

Two effective Approach for handwritten digit image recognition methods are executed which is based on Deep learning techniques Deep neural network and Convolutional neural network. Extensive comparing experiments on the MNIST database validate the effectiveness and superiority for the method. Future work includes more study on deep learning and optimizing algorithm.

A. Deep neural network

IMAGE ACQUISITION

Eliminating the unfavorable factors and filtering the characteristics of original handwritten digit images, we preprocess all images in the database. Images are transformed into binary images based on a selected threshold [2]. Adjustments are made to prevent serious deformation and the following steps are followed

- The ratio of width (W) to height (H) is calculated. If it exceeds a specific threshold then the image will be expanded to the size if HxH to prevent serious deformation of the digit.
- Calculate the ratio of the number of white pixels in all pixels, which is used to determine whether the image needs to deal with expansion and the related expansion coefficient.
- In the end, the digit skeleton is extracted to from the given data.

DATA PREPROCESSING

The primary goal of the pre-processing is to arrange the information to make character recognition process simpler.

As the considered database is noise free (clean), noise reduction techniques were not used. In real system, noise should remove. The following pre-processing steps have been applied.

1. Feature Scaling-- Mean normalization

The character shapes are always unique as they vary from person to person and instance to instance even if the same person writes. In order to make sure that the Machine learning algorithm deals these differences in the same way, normalization is used. An add on advantage is that the training process speeds up.

2. One Hot Encoding

One hot encoding is a data preprocessing stage in which categorical data is transformed into numerical form so that any machine learning algorithm can work on it [2]. Categorical data is generally stored in a variable which is usually a group of numbers. The possible values are often limited to a fixed set.

The categorical data here is the labels which are present in training data set which were extracted and represented using one hot encoding.

CLASSIFICATION USING DEEP NEURAL NETWORK

A neural network in which there is an input layer and an output layer, with more hidden layers in between is called a deep neural network. The number of nodes in input layer is determined by the number of features. There will be a bias node in each layer, to which input is not given, but it is connected to all other nodes in the next layer. Nodes contain activation functions like ReLu (Rectified Linear Unit) shown in Fig [2], sigmoid etc. The output layer nodes use ‘Softmax’ as activation function if a multi-layer classification is to be done. The basic architecture of a deep neural network is shown in the figure.

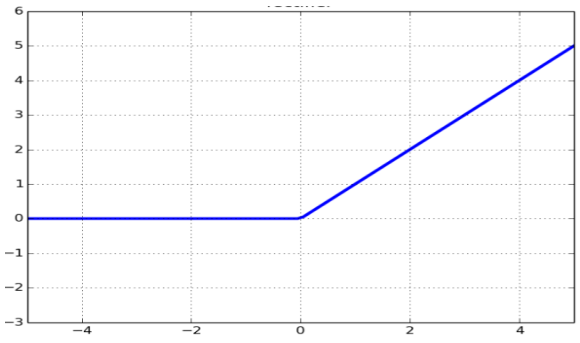

Figure 2:ReLu Activation.

A model with desired characteristics should be built and the normalized pixel values can be passed to train the model. The characteristics of the developed deep neural network [5] are shown:

1. Input layer

Input layer with 784 nodes was created using tflearn commands.

2. Hidden layers

Three fully connected hidden layers in which the last hidden layer has 32 nodes were created to make sure that the accuracy is high. 'ReLu' is used as the activation function since we are using Linear regression for classifying the digits between 0 and 9.

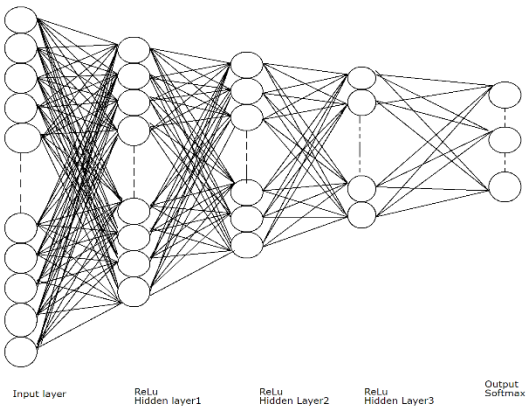

Figure 3: DNN Representation.

3. Output layer

'Softmax' was used as the activation function. Softmax is a probability distribution function which distributes the probability of a sample being classified as each of the digits. So, the output of this layer would be an n-dimensional array whose elements sum is 1. The predicted value would be the index value of the maximum argument of the array.

4. Back propagation

After one forward step, the weights should alter based on the local minima. This was done by Stochastic gradient descent (SGD). So, the back propagation was carried out using SGD.

The normalized inputs were passed to the DNN and the model was trained with batch size as 100, Validation set as 0.1 of whole set.

B. Convolution neural network (CNN):

Convolution neural network (CNN) is a specific type of Deep neural network which is useful for image recognition, object recognition etc. CNN use convolutional and pooling layers, which reflect the translation-invariant nature of most images. The usage of images will result in certain properties which reduces the number of parameters and increase the efficiency. CNN's architecture usually consists of three main layers which are Convolution layer, Pooling layer and Fully connected layer.

The Convolution (Conv) layer consists of a learnable filter which convolves over the image producing a dot product of the filter and respective portion of the image on which it is convolving. The resultant is a two-dimensional activation map, which is activated only when there is any change in that portion of the image. These features may be edges, lines, color blocks to circular, honeycomb pattern depending on the layer [1]. The filter which convolves has constant weights and biases associated with for the entire image. This layer is usually associated with a nonlinear activation function (most common is ReLu).

Pooling Layer is usually associated with Conv layer and is also similar to its working except that the filter convolves with bigger steps and different statistical operations (usually max). The purpose of this layer is to make the network impervious to scale and orientation changes in the image as well as decrease the number of parameters [2].

Fully connected layer connects to all the activations in the previous layer like an Artificial Neural Network layer.

IMPLEMENTATION:

We built a CNN model to recognize handwritten digits as another solution, to develop our understanding on this topic. As mentioned in the previous solution, the input is 784 pixels which is taken from 28X28 image. This is converted into an n-dimensional array of (I_n, 28, 28, 1) where I_n is the number of images (pixel

information) in the training data. This data has been used as an input for various CNNs. The implementation of the CNNs were done using Keras with Tensorflow as backend. Since we are dealing with grayscale images, we will be using two-dimensional Convolutional layers for all the solutions. The following is the architecture of our best performing CNN with an accuracy of 0.9864 (98.64%):

1) Convolution layer (filters=16)
   Dropout (fraction=0.1)
   Max Pooling layer (pool size=2)
2) Convolution layer (filters=32)
   Max Pooling layer (pool size=2)
   Dropout (fraction=0.2)
3) Convolution layer (filters=64)
   Dropout (fraction=0.3)
   Global average Pooling (pool size=2)
4) Dense (activation=RELU, 500 neurons)
   Dropout (fraction=0.4)
5) Dense (activation=Softmax, 10 neurons)

Where the filter size is three by three for all the Conv layers. Dens layer is the Fully connected layer. The last dense layer with Softmax activation is similar to that explained in the previous solution. The same has be depicted in the Fig [4].

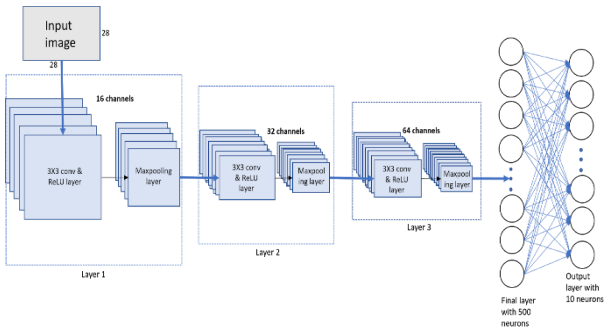

Figure 4: CNN Representation.

The main advantage in using a CNN over DNN is that the number of parameters are considerably reduced. Following are the number of parameters that are used for in the above CNN.

Conv layer 1 (filter size =3X3, filters = 16) = {(no of weights in filter) * depth + bias} * filters = {(3*3)*1+ 1}*16 = 160

Similarly:

Conv layer 2 (filters = 32) = 4640

Conv layer 3 (filters = 32) = 18496

Dense layer (500 neurons)
= {64 (channels)+1 (bias)} * 500 = 32,500

Dense layer (10 neurons) = 5010

Consider the first layer of the CNN which has 160 parameters, for the same input for the previous solution of ANN where the first layer has 129 neurons the number of parameters used is {(784 (input) * 128 (weights) + 128(biases)}= 100,480. this difference in number of parameters still increases with the increase in number of layers.

Since the main goal of our project is to understand the concepts behind image processing and working principles of the solutions, we have designed and trained the CNN with different layers combinations. The following are the basic overviews of their architectures:

1) One Conv layer with max and global average pooling.
2) Two Conv layers with max pooling for each layer of Conv.
3) Three Conv layers with max pooling.
4) Three Conv layers like above but the fully connected layer with 100 neurons.
5) Three Conv layer with no Dropout layers

The results of these layers have been discussed in detailed in the next chapter.

As shown previously, the number of parameters are considerably reduced compared to the ANN. This helps in both the computational costs as well as in avoiding overfitting.

IV. Results

We implemented handwritten digit recognizer using two deep learning techniques DNN (Deep neural network) and CNN (Convolutional Neural Network). The following are the results and the analysis on these techniques.

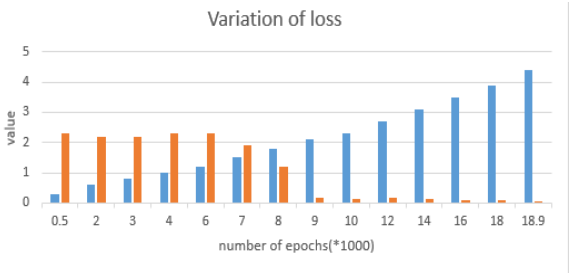

Figure 5: Variation loss in DNN.

DNN (Deep neural network):

In DNN method, we can observe that there is maximum loss in first epoch which reduces as the number of epochs increase. Below is the Fig. [5] which shows the

variation of total loss and training time with respective to epochs.

C. CNN (Convolutional Neural Network):

As mentioned in the previous chapter we have used five different CNN models for obtaining better accuracy. Model with 3 convolution layers had the highest had the highest accuracy in predicting the digits among all other models, while model with 1 conv layer had the lowest accuracy. The following are the results of the model's accuracy and loss values.

Type	Accuracy	Loss	Val_Accuracy	Val_Loss
3 convolution	98.64	4.26	97.95	5.79
Dense 100	98.38	4.88	98.1	7.68
No dropout	99.13	2.89	97.21	9.2
2 layer	96.84	10.18	93.61	21.79
1 layer	66.74	94.28	65.52	97.13

Table 1: Accuracy and Loss Values of CNN models

The Graph in the figure 6 plots the model values in to a bar graph representing in variation of the accuracy and loss values for the various models.

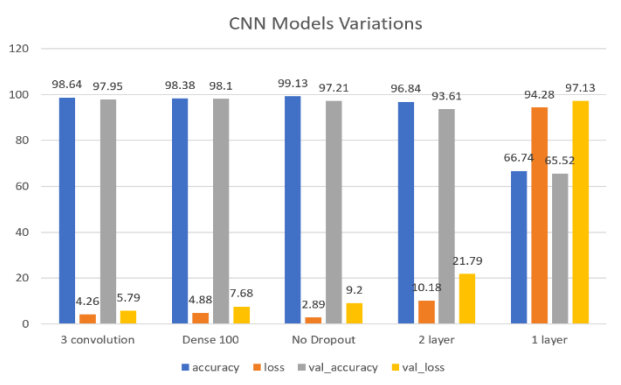

Figure 6: Accuracy and loss representation of CNN models

V. Conclusion

In this project Digit Recognizer, we successfully built two models based on **Deep neural network** (using tensorflow) and **Convolutional Neural Network** (using keras on tensorflow). The accuracy of DNN and CNN models are 98.5% and 98.7% respectively, which showcases clearly that both the algorithms predict numbers competitively. This implementation can be further refined to achieve higher accuracy using complex models of Neural Networks.

VI. References

[1] R. S. C. PRIYA, "HANDWRITTEN DIGIT RECOGNITION BY PROXIMAL SUPPORT VECTOR MACHINE," *JETIR* , vol. 4, no. 04, pp. 251-254, 2017.

[2] R. A. Parth Sane, "Pixel Normalization from Numeric Data as Input to Neural Networks," in *IEEE WiSPNET 2017 conference*, chennai, 2017.

[3] C. G. a. F. Berkhahn, "Entity Embeddings of Categorical Variables," Neokami Inc, 2016.

[4] S. S. C. Ayush Purohit, "A Literature Survey on Handwritten Character Recognition," *International Journal of Computer Science*, vol. 7, no. 1, pp. 1-5, 2016.

[5] M. Y. Y. K. Sigeru Omatu, "Bank note classification using neural networks," in *Emerging Technologies and Factory Automation*, NA, 2007.

[6] NA, "Convolutional Neural Networks for Visual Recognition," Stanford, [Online]. Available: http://cs231n.github.io/convolutional-networks/. [Accessed 1 dec 2018].

[7] Andy, "Convolutional Neural Networks Tutorial in TensorFlow," adventuresinmachinelearning, 27 april 2017. [Online]. Available: http://adventuresinmachinelearning.com/convolutional-neural-networks-tutorial-tensorflow/. [Accessed 1 dec 2018].