

**Hand-Written Digits Recognization**

Submitted to: Yogendra Sir

Submitted by: Vikram Sigh Gurjar &

Rahul Kumar

TABLE OF CONTENT

* Certificate
* Acknowledgement
* Abstract
* List of figures
* Introduction
* Theory
* Methodology
* Data Pre-processing
* Libraries Used
* CNN Model Used
* Data visualization
* Result Analysis
* Annexure



This is to certify that Mr. **Vikram Singh Gurjar** student of 3rd Year from Department of Computer Science, Rajasthan Technical University kota has undergone a Project work from June 3, 2019 to July 25, 2019 in **Deep Learning & Machine Learning** **Titled** “**MNIST:HAND-WRITTEN DIGITS PREDICTION**”



\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Project Incharge

Seal

PAGE 2

**Certificate**

Date: 25/07/19

This is to certify that Mr. **Rahul Kumar**  student of 3rd Year from Department of Computer Science, Rajasthan Techinical University kota has undergone a Project work from June 3, 2019 to July 25, 2019 in **Deep Learning & Machine Learning Titled-“MNIST:HAND-WRITTEN DIGITS PREDICTION”**

\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Project Incharge

Seal

PAGE 3

Acknowledgement

For making this project we would like to thank our course instructor, Yogendra Sir, he has helped us in learning the various concepts of the Machine Learning. Here we would also like to thank Kunal Sir and Rohit Sir for mentoring us throughout this project.



PAGE 4

Abstract

Rapid growth of data comes with a challenge of sorting and analyzing them, where raw data exists in graphical form, textual form or in images. Data science and machine learning finds its application in various fields like stock market, recommendation systems, image processing, aerial photography, military, weather forecasting etc.

This report is about our project on “Predicting Hand Written Digits” that addresses about data pre-processing and post processing which includes plotting, classification and prediction of Hand Written Digits appearance in real time and the ability of machine learning algorithms to deal with different set of data. In this project, we have tackled a classification problem of predicting where the Mnist digits will appear by accessing several data variable like Diiferent Types of HandWritten Digit images, etc. We have Train and tested Digit Images Using CNN(Convolutional Neural Network) Deep-Learning determine the results. In addition to this, we have also made use of several libraries to plot the data points on the map.



PAGE 5

List Of Figures

Fig. 1

Fig. 2

Fig. 3

Fig. 4

Fig. 5

Fig. 6

Fig. 7

Fig. 8

Data Science

Convolution

Relu

MaxPooling

Flatten

Input-Layer

Hidden-Layer

Output-Layer



PAGE 6

Introduction

In a computer vision system, handwritten digits recognition is a complex task that is central to a variety of emerging applications. It has been widely used by machine learning and computer vision researchers for implementing practical applications like computerized bank check numbers reading. In this study, we implemented a multi-layer fully connected neural network with one hidden layer for handwritten digits recognition. The testing has been conducted from publicly available MNIST handwritten database. From the MNIST database, we extracted 60,000 digits images for training and 10,000 digits images for performing the test. Our multi-layer artificial neural network has an accuracy of 99.60% with test performance.

The program I implement will mainly focus on identifying 0-9 from segmented pictures of handwritten digits. The input of my program is a gray level image, the intensity level of which varies from 0 to 255. For simplicity, input images are pre-treated to be of certain fixed size, and each input image should contain only one unknown digit in the middle. These requirements are not too harsh because they can be achieved using simple image processing or computer vision techniques. In addition, such pre-treated image data set are easy to obtain. In my implemen-tation, the popular MNIST data set ([1]) is a good choice. Each image in MNIST is already normalized to 28x28 in the above sense and the data set itself is publicly available. The MNIST

data set is really a huge one: it contains 60000 training samples and 10000 test samples. And it has become a standard data set for testing various algorithms.

Theory

Data science is a "concept to unify statistics, data analysis and their related methods" in order to "understand

and analyze actual phenomena" with data.[3] It employs techniques and theories drawn from many fields

within the broad areas of mathematics, statistics, information science, and computer science, in particular from

the subdomains of machine learning, classification, cluster analysis, data mining, databases, and visualization.

Data science – discovery of data insight

This aspect of data science is all about uncovering findings from data. Diving in at a granular level to mine and understand complex behaviors, trends, and inferences. It's about surfacing hidden insight that can help enable companies to make smarter business decisions.

For example:



Netflix data mines movie-viewing patterns to understand what drives user interest, and uses that to make decisions on which Netflix original series to produce

Data science – development of data product

A "data product" is a technical asset that: (1) utilizes data as input, and (2) processes that data to return algorithmically generated results. The classic example of a data product is a recommendation engine, which ingests user data, and makes personalized recommendations based on that data.

For example:

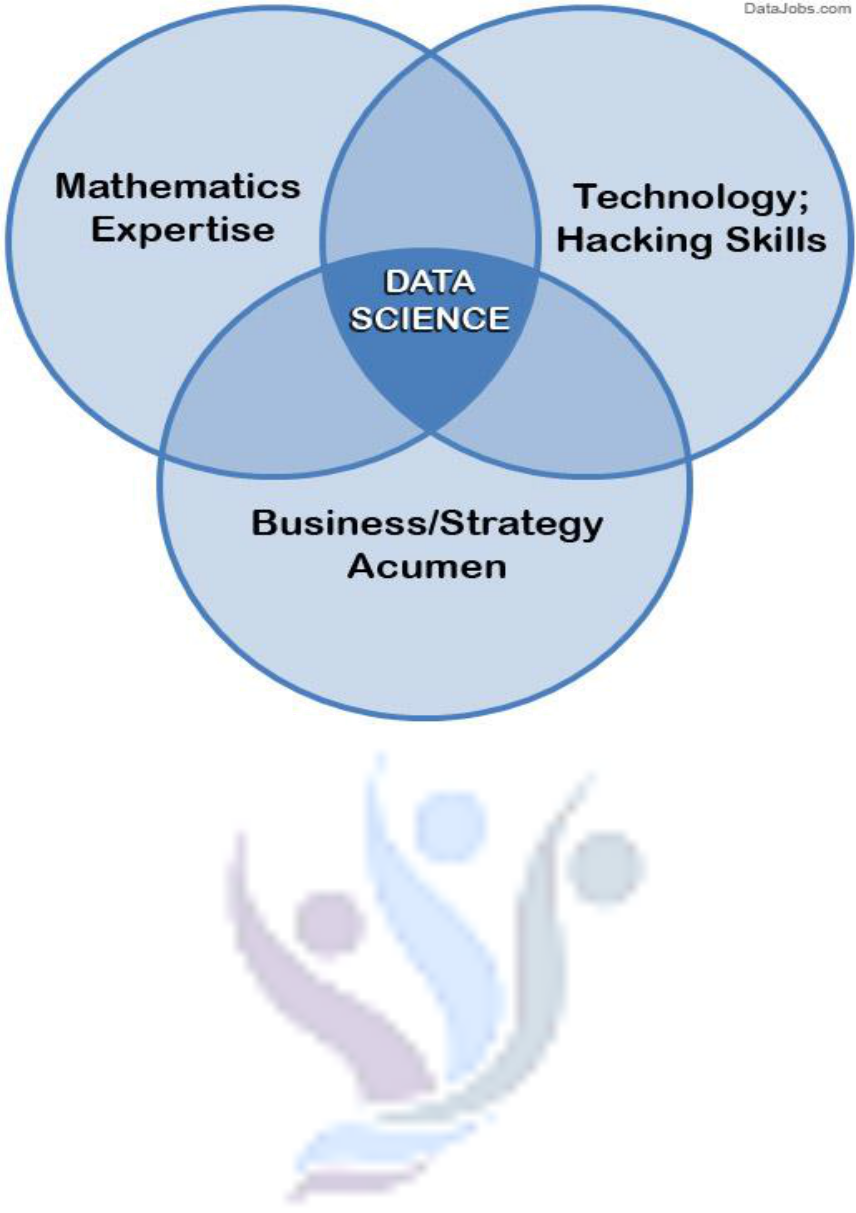
Amazon's recommendation engines suggest items for you to buy, determined by their algorithms. Netflix recommends movies to you. Spotify recommends music to you.

Machine learning and statistics are part of data science. The word learning in machine learning means that the

algorithms depend on some data, used as a training set, to fine-tune some model or algorithm parameters. This

encompasses many techniques such as regression, naive Bayes or supervised clustering.

PAGE 8



**Fig. 1 Data Science**

Supervised and unsupervised learning describe two ways in which machines algorithms can be set loose on a data set and expected to learn something useful from it.

**Supervised:**

If we are training our machine-learning task for every input with corresponding target, it is called [supervised](https://en.wikipedia.org/wiki/Supervised_learning) [learning,](https://en.wikipedia.org/wiki/Supervised_learning) which will be able to provide target for any new input after sufficient training. Our learning algorithm seeks a function from inputs to the respective targets. If the targets are expressed in some classes, it is called *classification* problem. Alternatively, if the target space is continuous, it is called *regression* problem.

* **Regression** analysis is widely used for prediction and forecasting, where its use has substantial overlapwith the field of machine learning. Regression analysis is also used to understand which among the independent variables are related to the dependent variable, and to explore the forms of these relationships. In restricted circumstances, regression analysis can be used to infer causal relationships between the independent and dependent variables.
* **Classification** model attempts to draw some conclusion from observed values. Given one or more inputsa classification model will try to predict the value of one or more outcomes. Outcomes are labels that can be applied to a dataset.

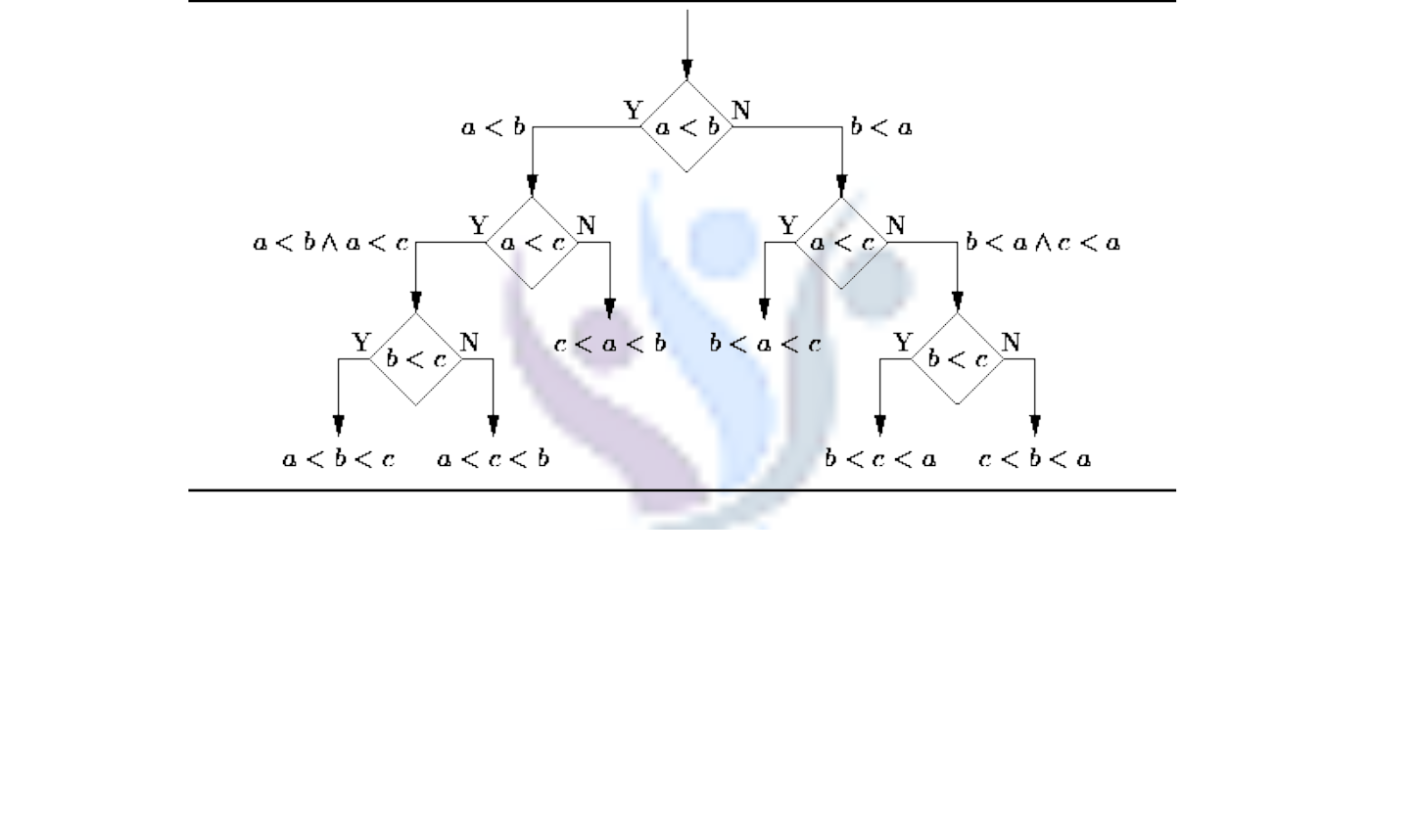
PAGE 9

**Unsupervised:** If we are training our machine-learning task only with a set of inputs, it is called unsupervisedlearning, which will be able to find the structure or relationships between different inputs. Most important unsupervised learning is *clustering*, which will create different cluster of inputs and will be able to put any new input in appropriate cluster.

* **Cluster** analysis or clustering is the task of grouping a set of objects in such a way that objects in thesame group (called a cluster) are more similar (in some sense or another) to each other than to those in

other groups (clusters). It is a main task of exploratory [data mining,](https://en.wikipedia.org/wiki/Data_mining) and a common technique for [statistical](https://en.wikipedia.org/wiki/Statistics) [data analysis,](https://en.wikipedia.org/wiki/Data_analysis) used in many fields, including [machine learning,](https://en.wikipedia.org/wiki/Machine_learning) [pattern recognition,](https://en.wikipedia.org/wiki/Pattern_recognition) [image](https://en.wikipedia.org/wiki/Image_analysis) [analysis,](https://en.wikipedia.org/wiki/Image_analysis) [information retrieval,](https://en.wikipedia.org/wiki/Information_retrieval) [bioinformatics,](https://en.wikipedia.org/wiki/Bioinformatics) [data compression,](https://en.wikipedia.org/wiki/Data_compression) and [computer graphics.](https://en.wikipedia.org/wiki/Computer_graphics)

1. **Decision Trees:** A decision tree is a decision support tool that uses a tree-like graph or model ofdecisions and their possible consequences, including chance-event outcomes, resource costs, and utility.

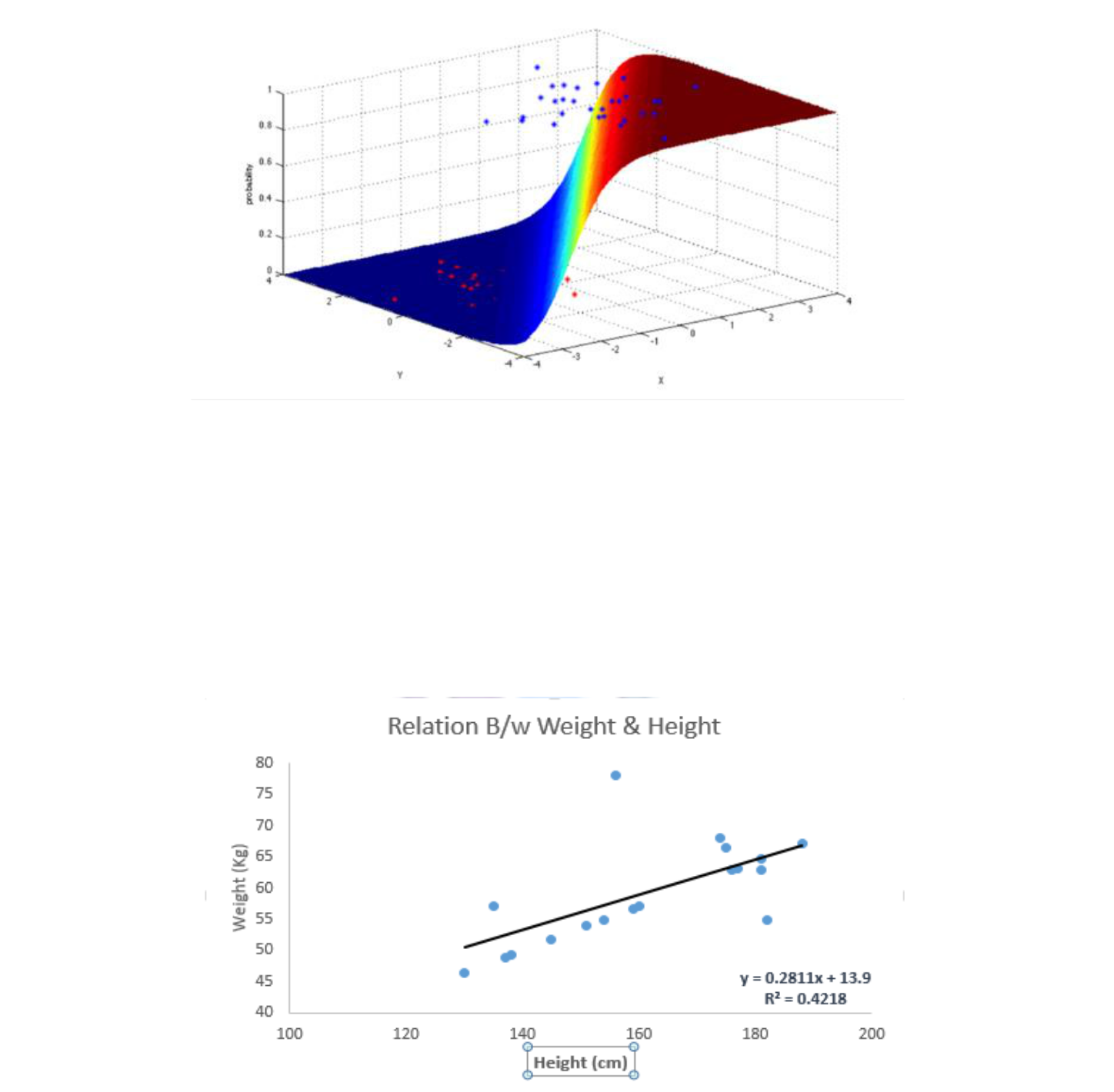


**Fig. 2 Decision Tree**

From a business decision point of view, a decision tree is the minimum number of yes/no questions that one has to ask, to assess the probability of making a correct decision, most of the time. As a method, it allows you to approach the problem in a structured and systematic way to arrive at a logical conclusion.

1. **Logistic Regression:** Logistic regression is a powerful statistical way of modeling a binomial outcome withone or more explanatory variables. It measures the relationship between the categorical dependent variable and one or more independent variables by estimating probabilities using a logistic function, which is the cumulative logistic distribution.

PAGE 10



**Fig. 3 Logistic Regression**

**3. Linear Regression**

It is used to estimate real values (cost of houses, number of calls, total sales etc.) based on continuous variable(s). Here, we establish relationship between independent and dependent variables by fitting a best line. This best fit line is known as regression line and represented by a linear equation Y= a \*X + b.

**Fig. Linear Regression**

**4. KNN (K- Nearest Neighbors)**

It is also a lazy algorithm. What this means is that it does not use the training data points to do any generalization. K nearest neighbors is a simple algorithm that stores all available cases and classifies new cases by a majority vote of its k neighbors. The case being assigned to the class is most common amongst its K

PAGE 11

nearest neighbors measured by a distance function. These distance functions can be Euclidean, Manhattan, Minkowski and Hamming distance. First three functions are used for continuous function and fourth one (Hamming) for categorical variables. If K = 1, then the case is simply assigned to the class of its nearest neighbor. At times, choosing K turns out to be a challenge while performing KNN modeling.



**Fig. 4 KNN**

CNN Model

## Table of contents

* What is CNN ?
* Why should we use CNN ?
* Few Definitions
* Layers in CNN
* Keras Implementation

# 1. What is CNN ?

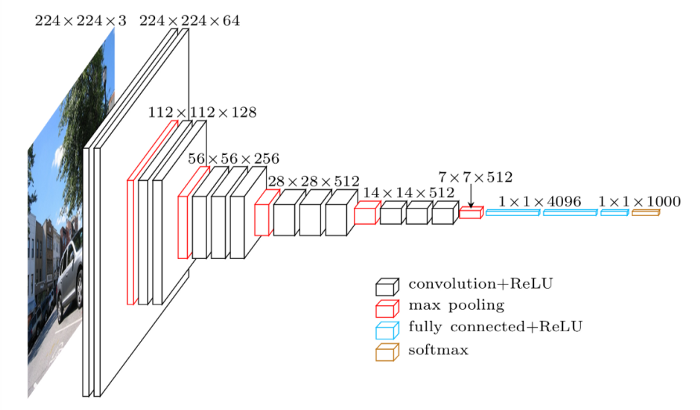
Computer vision is evolving rapidly day-by-day. Its one of the reason is deep learning. When we talk about computer vision, a term convolutional neural network( abbreviated as CNN) comes in our mind because CNN is heavily used here. Examples of CNN in computer vision are face recognition, image classification etc. It is similar to the basic neural network. CNN also have learnable parameter like neural network i.e, weights, biases etc.

# 2. Why should we use CNN ?

## Problem with Feedforward Neural Network

Suppose you are working with MNIST dataset, you know each image in MNIST is 28 x 28 x 1(black & white image contains only 1 channel). Total number of neurons in input layer will 28 x 28 = 784, this can be manageable. What if the size of image is 1000 x 1000 which means you need 10⁶ neurons in input layer. Oh! This seems a huge number of neurons are required for operation. It is computationally ineffective right. So here comes Convolutional Neural Network or CNN. . In the following example you can see that initial the size of the image is 224 x 224 x 3. If you proceed without convolution then you need 224 x 224 x 3 = 100, 352 numbers of neurons in input layer but after applying convolution you input tensor dimension is reduced to 1 x 1 x 1000. It means you only need 1000 neurons in first layer of feedforward neural network.





**3. Few Definitation**

## 3.1 Image Representation

Thinking about images, its easy to understand that it has a height and width, so it would make sense to represent the information contained in it with a two dimensional structure (a matrix) until you remember that images have colors, and to add information about the colors, we need another dimension, and that is when Tensors become particularly helpful.



fig.RGB representation of a image

Images are encoded into color channels, the image data is represented into each color intensity in a color channel at a given point, the most common one being RGB, which means Red, Blue and Green. The information contained into an image is the intensity of each channel color into the width and height of the image, just like this

**3.2 Edge Detection**

Every image has vertical and horizontal edges which actually combining to form a image. Convolution operation is used with some filters for detecting edges. Suppose you have gray scale image with dimension 6 x 6 and filter of dimension 3 x 3(say). When 6 x 6 grey scale image convolve with 3 x 3 filter, we get 4 x 4 image. First of all 3 x 3 filter matrix get multiplied with first 3 x 3 size of our grey scale image, then we shift one column right up to end , after that we shift one row and so on.

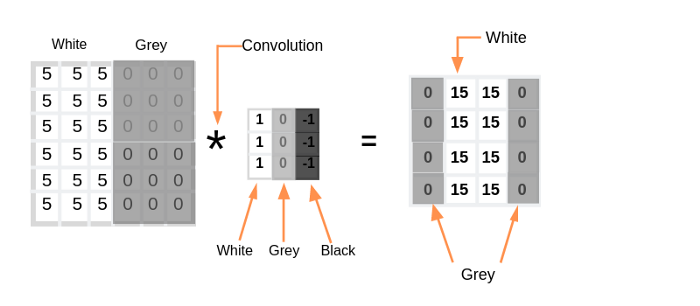


Fig 1.Convolution operation

The convolution operation can be visualized in the following way. Here our image dimension is 4 x 4 and filter is 3 x 3, hence we are getting output after convolution is 2 x 2.

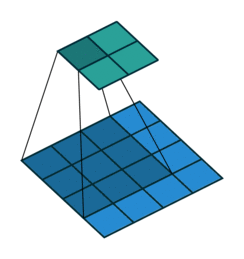


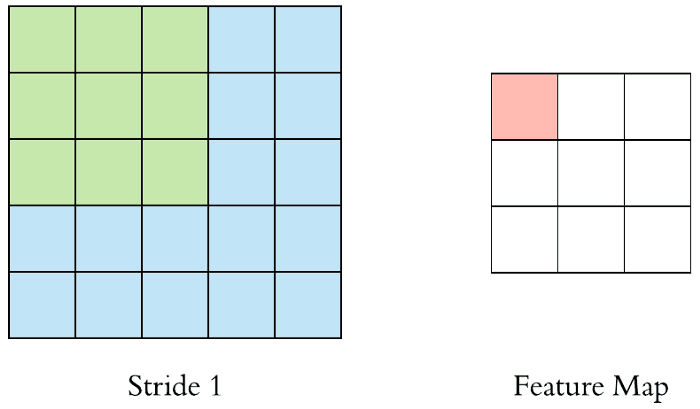
fig 2.Visualization of convolution

If we have N x N image size and F x F filter size then after convolution result will be

(N x N) \* (F x F) = (N-F+1)x(N-F+1)(Apply this for above case)

## 3.3 Stride and Padding

Stride denotes how many steps we are moving in each steps in convolution.By default it is one.



We can observe that the size of output is smaller that input. To maintain the dimension of output as in input , we use padding. Padding is a process of adding zeros to the input matrix symmetrically. In the following example,the extra grey blocks denote the padding. It is used to make the dimension of output same as input.

Let say ‘p’ is the padding

Initially(without padding)

(N x N) \* (F x F) = (N-F+1)x(N-F+1)---(1)

After applying padding

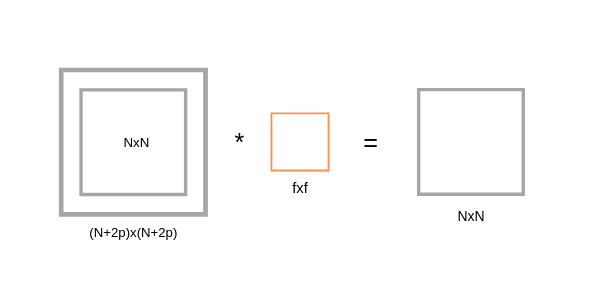


fig 3.After applying padding

If we apply filter F x F in (N+2p) x (N+2p) input matrix with padding, then we will get output matrix dimension (N+2p-F+1) x (N+2p-F+1). As we know that after applying padding we will get the same dimension as original input dimension (N x N). Hence we have,

(N+2p-F+1)x(N+2p-F+1) equivalent to NxN

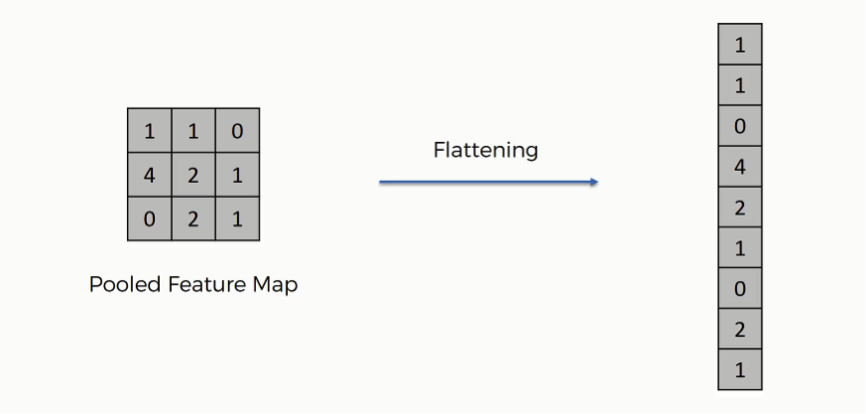
N+2p-F+1 = N ---(2)

p = (F-1)/2 ---(3)

The equation (3) clearly shows that Padding depends on the dimension of filter.

**3.4 Flatten**

Flatten is the function that converts the pooled feature map to a single column that is passed to the fully connected layer. Dense adds the fully connected layer to the **neural network**.



# 4. Layers in CNN

There are five different layers in CNN

* Input layer
* Convo layer (Convo + ReLU)
* Pooling layer
* Fully connected(FC) layer
* Softmax/logistic layer
* Output layer

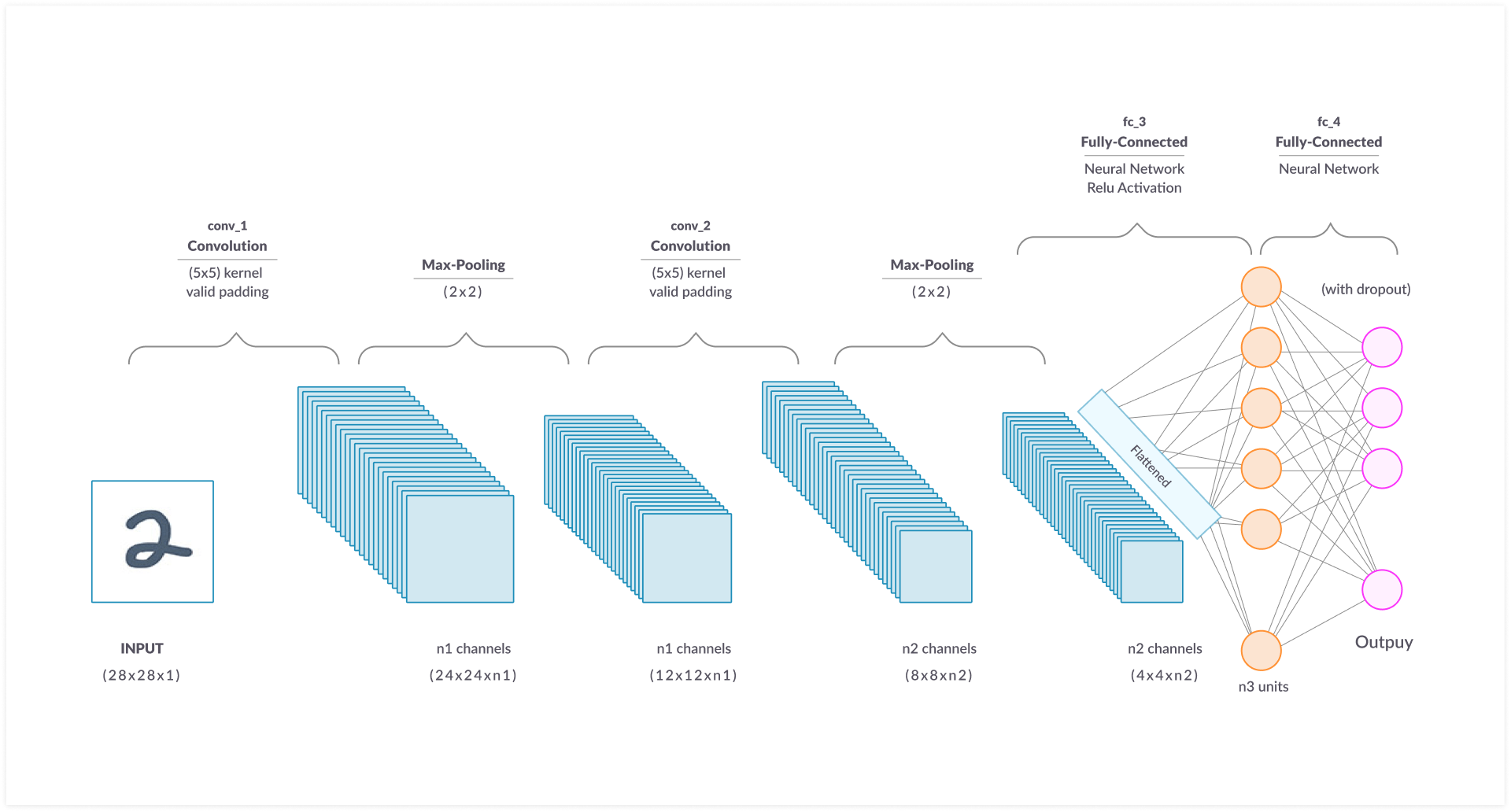


fig 4. Convolutional neural network

Methodology

To understand the methodology adopted, we first understand our dataset and the variables.

**1. Dataset**

Our MNIST Dataset is already normalized to 28x28 in the above sense and the data set itself is publicly available.The MNIST dataset is really a huge one it contains 60000 training samples and 10000 test samples

**2. Features :**

* There are 60,000 HandWritten Gray-Scale Images Works as a Features

3. Cleaning the data:

Cleaning up the dataset is the most crucial step before feeding it to the learning algorithm. The data generated in real time makes no promise to be the way it is required because different minds generate different data. In this project, the dataset used is a standard one, well cleaned and up to the mark, So we need not to work on that. But for other cases, cleaning is the way.

4. Converting the data into the model compatible shape:

The data needs to be converted to the model compatible shape before moving further.

train\_images = train\_images.reshape((60000, 28, 28, 1))

5. Feature Scaling:

Converting the features into scaled values is a really helpful option to improve the accuracy of your model. In our case, converting did not require any complex operation but only dividing the pixel intensities by 255 worked because If we see the mathematical formula for MinMax scaling it’s the same thing actually. The labels for every image in the data set was in the form of an integer value. These values were converted into one hot encoding format because we had 10 different classes and the model gives 10 different probability values each corresponding to a different label. The model gives that class as the output which has the greatest probability.

train\_images = train\_images.astype('float32') / 255

test\_images = test\_images.astype('float32') / 255

# **6. Designing the CNN model:**

First we need to take a container for the layers (Sequential in our case). Then we add layers to it. The

data while training is processed at different layers and the output is passed to the other layer.

Apply 32 Convolutional with (3,3) filter and use (2,2) matrix for maxpooling this process repeat

2 Times for accurate model and last apply flatten and it is the input for first hidden Layers

# **7 .Compile Fit CNN Model:**

"""COMPILING THE MODEL"""

model.compile(optimizer='rmsprop',loss='categorical\_crossentropy',metrics=['accuracy'])

we have done Filtering, Max Pooling, Relu and Flattening. These are steps of training of a CNN model.

These are common terminologies

## 8.Fit CNN Model:

Fitting the model means making it learn the data and relations between the features.

""" TRAINING THE MODEL"""

model.fit(train\_images, train\_labels, epochs=50, batch\_size=64)

9. Storing the trained model into an H5 And Pickle file:

With great computation power, a deep learning model also requires a considerably high amount of time to be

trained. Because of this reason, it is impractical to train the model again and again for every time the server

functions. So to reduce the time taking, the model was saved in an **H5** and **Pickle** file which stores only the

details of the model in JSON format. The details stored by this file are enough for making a copy of the pre-

trained model even in a blink of an eye.

import pickle

# Dump the Trained MNIST Data with Pickle

MNIST\_pkl\_filename = 'digit\_predict.pkl'

# Open the file to save as pkl file

MNIST\_model\_pkl = open(MNIST\_pkl\_filename, 'wb')

pickle.dump(model, MNIST\_model\_pkl)

MNIST\_model\_pkl.close()

Data Pre-processing

For pre-processing, we considered Python as our options for the project

MNIST dataset contains one row for each of the 60,000 training instances and one column for each of the 784 pixels in a 28 x 28 image (60,000,28,28). The data as downloaded doesn't have column labels but are arranged as (60,000,28,28,1) "row 1 column 1, row 1 column 2, row 1 column 3..." and so on. This is a useful enough representation for machine learning.

After the reshaping features column need to be rescalling because range of pixel (0-255)

Label Encode of MNIST Labels Because Label contains (0-9) digits this is done by using Keras Library **to\_categorical**



Libraries used

* numpy
* pandas
* matplotlib.pyplot
* Tensorflow
* pickle
* Base64
* Keras

o from keras import layers

o from keras import models

o from keras import to\_categorical

* mnist
* flask

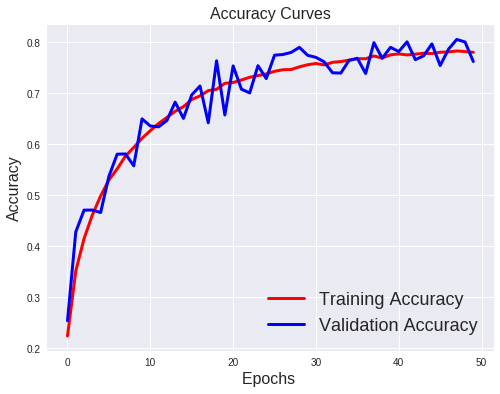
Data visualization

**Accuracy and Loss:-**

Test accuracy 99%+ implies the model is trained well for prediction. If we visualize the whole training log, then with more number of epochs the loss and accuracy of the model on training and testing data converged thus making the model a stable one.

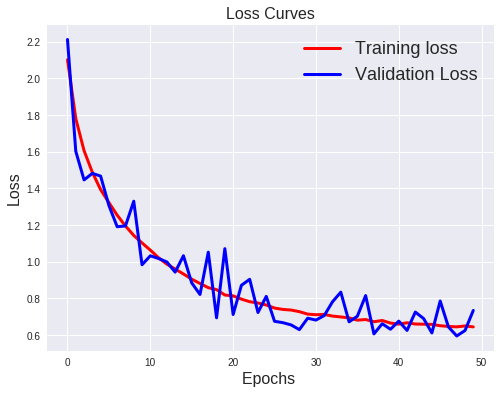
**Model-Accuracy :-**

when no of epochs is increase then accuracy-score also increase



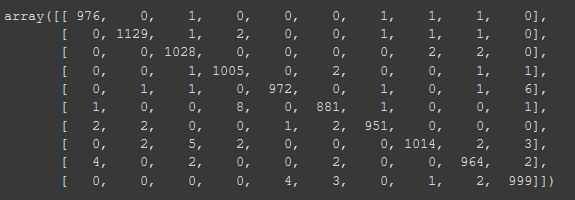
**Model-Loss:-**

when no of epochs is increase then Los-score also Decrease

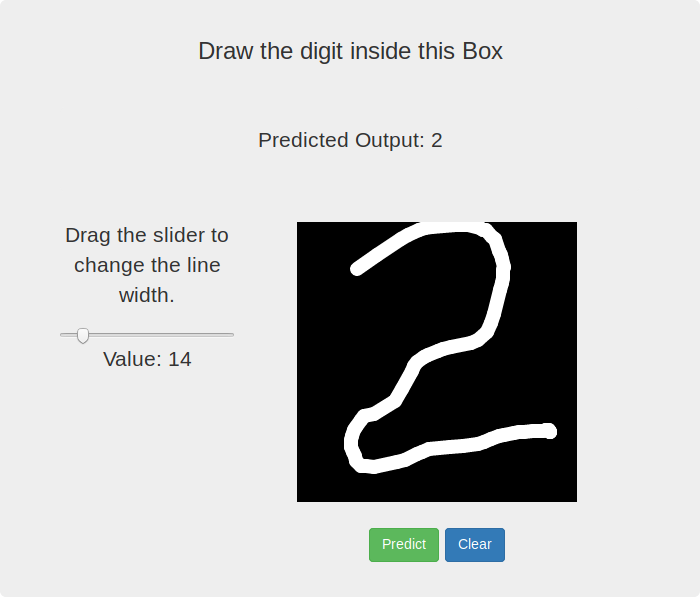
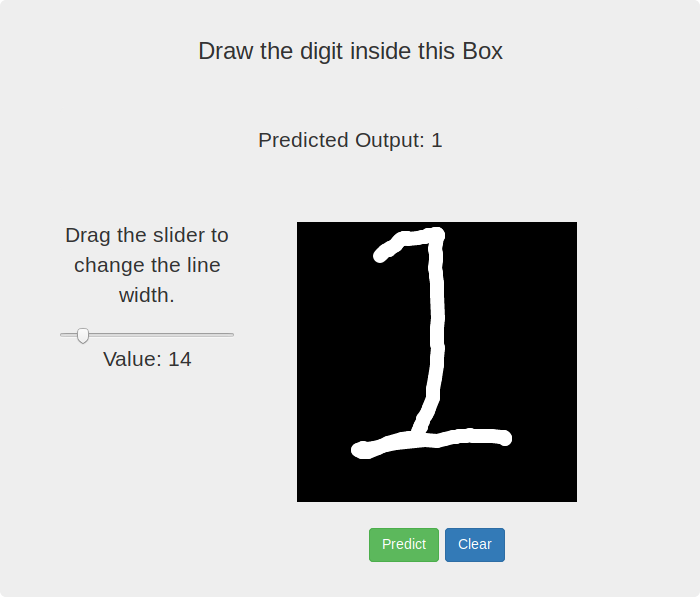


**Confusion-matrix:-**

We are ready now to code this into Python. The following code shows a confusion matrix for a multi-class machine learning problem with ten labels, so for example an algorithms for recognizing the ten digits from handwritten characters.



Result Analysis

****

Conclusion

Here in this project we were able to correctly visualize the MNIST Handwritten Digit Prediction. We were able to correctly Identify the Digit based on CNN Model although accuracy score of CNN Model is good,We were able to correctly Visualize the Loss and Accuracy of the Train Model

We were also able to make the predictions of the Test Images which is given by users and

note that 98% Gives accurate result of my CNN Model



PAGE 20

Annexure

**Code for visualization :-**

import pandas as pd

import matplotlib.pyplot as plt

from mpl\_toolkits.basemap import Basemap

plt.figure(1, figsize=(20,10))



m1 = Basemap(projection='merc', llcrnrlat=-60, urcrnrlat=65, llcrnrlon=-180, urcrnrlon=180, lat\_ts=0, resolution='c')

m1.fillcontinents(color='#cc9966',lake\_color='#99ffff')

m1.drawmapboundary(fill\_color='#99ffff')

m1.drawcountries(linewidth=0.1, color="w")

x, y = m1(df.longitude.tolist(),df.latitude.tolist())

m1.scatter(x,y, s=3, c="red", lw=0, zorder=5)

**Code for training the data with Logistic Regression**

from sklearn.linear\_model import LogisticRegression

clas = LogisticRegression(random\_state = 0)

clas1 = LogisticRegression(random\_state = 0)

print 'Training data with latitudes...'

clas.fit(X\_train, Y\_train)

print 'Training data with longitudes...'

PAGE 21

clas1.fit(X\_train, Y1\_train)

