Hand Written Digit Classification With

Machine Learning

A REPORT SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUREMENTS

FOR THE DEGREE OF

BACHELOR OF TECHNOLOGY

IN

INFORMATION TECHNOLOGY

Under the supervision of

Mr. Piyush Singh Bhushan

Head of Department, Department of Information Technology



By

Name – Himanshu Gupta

Roll No. – 2001640130022

Year – IT-3 Year(2022-23)

Pranveer Singh Institute Of Technology

DR. A.P.J ABDUL KALAM TECHNICAL UNIVERSITY

Lucknow

TABLE OF CONTENTS

Declaration………………………………………………………………………………………………II

Acknowledgement………………………………………………………………………………………..…III

Certificate……………………………………………………………………………………………..………….iv

Abstract……………………………………………………………………………………………….…………….v

Table of Contents…………………………………………………………………………..……………….VI

CHAPTER 1…………………………………………………………………………………………1 – 3

INTRODUCTION………………………………………………………………………………………..

* 1. Problem Statement………………………………………………………………………
  2. Problem Motivation…………………………………………………………………….
  3. Flow Chart Diagrams………………………………………………………………………..
  4. Data Flow Diagrams………………………………………………………………………..

CHAPTER 2………………………………………………………………………………………..4 – 6

PRELIMINARY STUDY……………………………………………………………………………

2.1 Problem Specifications………………………………………………………………………..

2.2 Literature review………………………………………………………………………………..

CHAPTER 3…………………………………………………………………………………………….7 -9

MODULES………………………………………………………………………………………………….

3.1 Module 1……………………………………………………………………………………………….

3.2 Module 2……………………………………………………………………………………………..

3.3 Module 3……………………………………………………………………………………………..

3.4 Module 4…………………………………………………………………………………………….

CHAPTER 4…………………………………………………………………………………………10 – 11

HARDWARE DESCRIPTION………………………………………………………………………

4.1 Anaconda Requirements……………………………………………………………………….

4.2 RAM Requirements………………………………………………………………………………

4.3 Library Requirements………………………………………………………………………….

4.4 Data Management………………………………………………………………………………..

CHAPTER 5……………………………………………………………………………………………………12 - 13

MODEL DESCRIPTION………………………………………………………………………………………..

5.1 Anaconda Navigator………………………………………………………………………………………

5.2 Jupyter Notebook…………………………………………………………………………………………..

5.3 Training & Test Datasets………………………………………………………………………………….

CHAPTER 6…………………………………………………………………………………………………14 – 25

CODING……………………………………………………………………………………………………………….

6.1 Library Inclusions…………………………………………………………………………………………….

6.2 Data Sets Inclusions…………………………………………………………………………………………

6.3 Data Mining & Data Science…………………………………………………………………………

MACHINE LEARNING ALGORITHMS………………………………………………………………..

6.4 Classification……………………………………………………………………………………………………..

6.5 Types of Classifications……………………………………………………………………………………..

6.6 KNN Classifer ………………………………………………………………………….

6.7 Convolution Neural Network for classification………………………………………………….

DATA FILTRATION WAYS………………………………………………………………………………..

6.9 Imputation………………………………………………………………………………………………….

DATA REPRESENTATION & VISUALIZATION…………………………………………………………

7.1 Graphical Representation……………………………………………………………………………………

MODEL EVALUATION…………………………………………………………………………………………………

8.1 Mean Absolute Error Evaluation…………………………………………………………………………..

CHAPTER 8………………………………………………………………………………………………………26- 30

SNAPSHOTS……………………………………………………………………………………………………………..

CHAPTER 9………………………………………………………………………………………………………..31 – 33

CONCLUSION and FUTURE SCOPE………………………………………………………………………..

REFERENCES………………………………………………………………………………………………………34- 36

Declaration

I, Himanshu Gupta hereby declare the project entitled Hand Written Digit Classification With Machine Learning being submitted towards the partial fulfillment of the requirements for the degree of Bachelor of Technology in Information Technology(IT) is a project work carried by me under the supervision of Mr. Piyush Singh BhushanHead of Department,Department of Information Technology, and have not been submitted anywhere else. I assert that statements made and conclusions drawn are an outcome of my research work.

I have followed the guidelines provided by the University in writing the Project report.

Name – Himanshu Gupta

Roll No. 2001640130022

Date:

Acknowledgement

I would like to share my sincere gratitude to all those who help me in completion of the project .During the work I faced many challenges due to lack of knowledge and experience but these people helped to get over form all the difficulties and in final completion of idea to a shaped sculpture.

I would like to thanks Mr. Piyush Singh sir for his governance and guidance, because of which I was able to learn the minute aspects and the parameters of the project.

I would like also to show my gratitude to our HOD sir Mr. Piyush Bhushan for his continuous help and monitoring during the project work.

In last I would like to thank Pranveer Singh Institute of Technology, for providing me such an opportunity to learn form these experiences.

Thank You All.

Date:-

Name – Himanshu Gupta

Roll No.:-2001640130022

Signature:-

Abstract

Handwritten character recognition is one of the practically important issues in pattern recognition applications. The main purpose of this project is to build an automatic handwritten digit recognition method for the recognition of handwritten digit strings. To accomplish the recognition task, first, the digits will be segmented into individual digits. Then, a digit recognition module is employed to classify each segmented digit completing the handwritten digit string recognition task. The applications of digit recognition include postal mail sorting, bank check processing, form data entry, etc. The heart of the problem lies within the ability to develop an efficient algorithm that can recognize handwritten digits and which is submitted by users by the way of a scanner, tablet, and other digital devices.

Certificate

This is to certify that the project titled “**Hand Written Digit Classification With Machine Learning**” has been submitted to the Department of Information Technology ,

Pranveer Institute of Technology(APJ Abdul Kalam Technical University , Lucknow) for the fulfillment of the requirement of the award of the degree of Bachelor Of Technology in “Information Technology” by following student of second year B.Tech.(Information Technology).

Himanshu Gupta

(2001640130022)

Problem Statement

The goal of this project is to create a model that will be able to recognize and determine the handwritten digits from its image by using the concepts of Convolution Neural Network. Though the goal is to create a model which can recognize the digits, it can be extended to letters and an individual’s handwriting. The major goal of the proposed system is understanding Convolutional Neural Network, and applying it to the handwritten recognition system.

Problem Motivation

Hand writing recognition of characters has been around since the 1980s.The task of handwritten digit recognition, using a classifier, has great importance and use such as – online handwriting recognition on computer tablets, recognize zip codes on mail for postal mail sorting, processing bank check amounts, numeric entries in forms filled up by hand (for example ‐ tax forms) and so on. There are different challenges faced while attempting to solve this problem. The handwritten digits are not always of the same size, thickness, or orientation and position relative to the margins. Our goal was to implement a pattern classification method to recognize the handwritten digits provided in the MINIST data set of images of hand written digits (0‐9). The data set used for our application is composed of 300 training images and 300 testing images, and is a subset of the MNIST data set [1] (originally composed of 60,000 training images and 10,000 testing images). Each image is a 28 x 28 grayscale (0‐255) labeled representation of an individual digit.

Flow Chart Diagram

Prediction

Model Evaluation

Model Construction

Training And Validation

PreProcessing

Data Flow Diagram

Data Obtained from Kaggle Data sets

Train the Following Neurons in Neural Network

Feed data to 28 Neurons

Make Images flatten to feed first layer

Apply Data Reduction Techniques

Problem Specifications

The given problem can be solved either by applying complicated mathematics by a human being but in today’s contrast as the technology is at it resonating point it is preferable to solve problem with ML model techniques which have accuracy and requires less time to get optimum solutions.

For the given problem we classify such problem as regression problem in terms of machine learning as in this particular we have to give values “features” as outcome not to classify prices as “high” or “low”.

Literature Review

An early notable attempt in the area of character recognition research is by Grimsdale in 1959. The origin of a great deal of research work in the early sixties was based on an approach known as analysisby-synthesis method suggested by Eden in 1968. The great importance of Eden's work was that he formally proved that all handwritten characters are formed by a finite number of schematic features, a point that was implicitly included in previous works. This notion was later used in all methods in syntactic (structural) approaches of character recognition.

**1. K. Gaurav, Bhatia P. K. ,** his paper deals with the various pre-processing techniques involved in the character recognition with different kind of images ranges from a simple handwritten form based documents and documents containing colored and complex background and varied intensities.In this, different preprocessing techniques like skew detection and correction, image enhancement techniques of contrast stretching, binarization, noise removal techniques, normalization and segmentation, morphological processing techniques are discussed.

**2. Sandhya Arora** , used four feature extraction techniques namely, intersection, shadow feature, chain code histogram and straight line fitting features. Shadow features are computed globally for character image while intersection features, chain code histogram features and line fitting features are computed by dividing the character image into different segments. On experimentation with a dataset of 4900 samples the overall recognition rate observed was 92.80% for Devanagari characters.

**3. Brakensiek, J. Rottland, A. Kosmala, J. Rigoll**, in their paper a system for off-line cursive handwriting recognition is described which is based on Hidden Markov Models (HMM) using discrete and hybrid modelling techniques. Handwriting recognition experiments using a discrete and two different hybrid approaches, which consist of a discrete and semi-continuous structures, are compared. It is found that the recognition rate performance can be improved of a hybrid modelling technique for HMMs, which depends on a neural vector quantizer (hybrid MMI), compared to discrete and hybrid HMMs, based on tired mixture structure (hybrid - TP), which may be caused by a relative small data set.

**4. R. Bajaj, L. Dey, S. Chaudhari** , employed three different kinds of features, namely, the density features, moment features and descriptive component features for classification of Devanagari Numerals. They proposed multi classifier connectionist architecture for increasing the recognition reliability and they obtained 89.6% accuracy for handwritten Devanagari numerals.

**5. G. Pirlo and D. Impedovo** in his work on , presented a new class of membership functions, which are called Fuzzymembership functions (FMFs), for zoning-based classification. These FMFs can be easily adapted to the specific characteristics of a classification problem in order to maximize classification performance. In this research, a realcoded genetic algorithm is presented to find, in a single optimization procedure, the optimal FMF, together with the optimal zoning described by Voronoi tessellation. The experimental results, which are carried out in the field of handwritten digit and character recognition, indicate that optimal FMF performs better than other membership functions based on abstract level, ranked-level, and measurement-level weighting models, which can be found in the literature.

**6. Sushree Sangita Patnaik and Anup Kumar Panda** May 2011 , this paper proposes the implementation of particle swarm optimization (PSO) and bacterial foraging optimization (BFO) algorithms which are intended for optimal harmonic compensation by minimizing the undesirable losses occurring inside the APF itself. The efficiency and effectiveness of the implementation of two approaches are compared for two different conditions of supply. The total harmonic distortion (THD) in the source current which is a measure of APF performance is reduced drastically to nearly 1% by employing BFO. The results demonstrate that BFO outperforms the conventional and PSO based approaches by ensuring excellent functionality of APF and quick prevail over harmonics in the source current even under unbalanced supply.

**7. M. Hanmandlu, O.V. Ramana Murthy** have presented in their study the recognition of handwritten Hindi and English numerals by representing them in the form of exponential membership functions which serve as a fuzzy model. The recognition is carried out by modifying the exponential membership functions fitted to the fuzzy sets. These fuzzy sets are derived from features consisting of normalized distances obtained using the Box approach. The membership function is modified by two structural parameters that are estimated by optimizing the entropy subject to the attainment of membership function to unity. The overall recognition rate is found to be 95% for Hindi numerals and 98.4% for English numerals.

**8. Renata F. P. Neves** have proposed SVM based offline handwritten digit recognition. Authors claim that SVM outperforms the Multilayer perceptron classifier. Experiment is 12 carried out on NIST SD19 standard dataset. Advantage of MLP is that it is able to segment non-linearly separable classes. However, MLP can easily fall into a region of local minimum, where the training will stop assuming it has achieved an optimal point in the error surface. Another hindrance is defining the best network architecture to solve the problem, considering the number of layers and the number of perceptron in each hidden layer. Because of these disadvantages, a digit recognizer using the MLP structure may not produce the desired low error rate.

Anaconda Requirements

It is primary and important to have Anaconda Navigator with version 3.8.3 for best model performance.

For download refer: https://docs.anaconda.com/anaconda/navigator/index.html

Second it should have Jupyter lab or Notebook in running status .

And last it should be already done completed with all library requirements

RAM REQUIREMENTS

The minimum ram requirements for this model is 2GB ,ie 2giga bytes is much sufficient for the model to efficiently perform.

Library Requirement

**Keras**

Keras is a powerful and easy-to-use free open source Python library for developing and evaluating deep learning models. 18 It wraps the efficient numerical computation libraries Theano and TensorFlow and allows you to define and train neural network models in just a few lines of code. It uses libraries such as Python, C#, C++ or standalone machine learning toolkits. Theano and TensorFlow are very powerful libraries but difficult to understand for creating neural networks. Keras is based on minimal structure that provides a clean and easy way to create deep learning models based on TensorFlow or Theano. Keras is designed to quickly define deep learning models. Well, Keras is an optimal choice for deep learning applications.

**TensorFlow**

TensorFlow is a Python library for fast numerical computing created and released by Google. It is a foundation library that can be used to create Deep Learning models directly or by using wrapper libraries that simplify the process built on top of TensorFlow. TensorFlow tutorial is designed for both beginners and professionals. Our tutorial provides all the basic and advanced concept of machine learning and deep learning concept such as deep neural network, image processing and sentiment analysis. TensorFlow is one of the famous deep learning frameworks, developed by Google Team. It is a free and open source software library and designed in Python programming language, this tutorial is designed in such a way that we can easily implements deep learning project on TensorFlow in an easy and efficient way. Unlike other numerical libraries intended for use in Deep Learning like Theano, TensorFlow was designed for use both in research and development and in production systems. It can run on single CPU systems, GPUs as well as mobile devices and largescale distributed systems of hundreds of machines.

MatPlot Lib

**Matplotlib** is a [plotting](https://en.wikipedia.org/wiki/Plotter) [library](https://en.wikipedia.org/wiki/Library_(computer_science)) for the [Python](https://en.wikipedia.org/wiki/Python_(programming_language)) programming language and its numerical mathematics extension [NumPy](https://en.wikipedia.org/wiki/NumPy). It provides an [object-oriented](https://en.wikipedia.org/wiki/Object-oriented_programming) [API](https://en.wikipedia.org/wiki/API) for embedding plots into applications using general-purpose [GUI toolkits](https://en.wikipedia.org/wiki/GUI_toolkit) like [Tkinter](https://en.wikipedia.org/wiki/Tkinter), [wxPython](https://en.wikipedia.org/wiki/WxPython), [Qt](https://en.wikipedia.org/wiki/Qt_(software)), or [GTK](https://en.wikipedia.org/wiki/GTK). There is also a [procedural](https://en.wikipedia.org/wiki/Procedural_programming) "pylab" interface based on a [state machine](https://en.wikipedia.org/wiki/State_machine) (like [OpenGL](https://en.wikipedia.org/wiki/OpenGL)), designed to closely resemble that of [MATLAB](https://en.wikipedia.org/wiki/MATLAB), though its use is discouraged. [SciPy](https://en.wikipedia.org/wiki/SciPy) makes use of Matplotlib.

Installation command for matplotlib

In windows pip install matplotlib

Numpy

NumPy, which stands for Numerical Python, is a library consisting of multidimensional array objects and a collection of routines for processing those arrays. Using NumPy, mathematical and logical operations on arrays can be performed.

NumPy is a Python package. It stands for ‘Numerical Python’. It is a library consisting of multidimensional array objects and a collection of routines for processing of array.

For Machine learning models it is primitive choice to give numerical data with least errors hence this library is key to perfection in this case.

Seaborn

**Seaborn**is a data visualization library built on top of matplotlib and closely integrated with pandas data structures in Python. Visualization is the central part of Seaborn which helps in exploration and understanding of data.

Seaborn offers the following functionalities:

1. Dataset oriented API to determine the relationship between variables.
2. Automatic estimation and plotting of linear regression plots.
3. It supports high-level abstractions for multi-plot grids.
4. Visualizing univariate and bivariate distribution.

Data Management

**Data Collection**

• The quantity & quality of your data dictate how accurate our model is

• The outcome of this step is generally a representation of data (Guo simplifies to specifying a table) which we will use for training

• Using pre-collected data, by way of datasets from Kaggle, UCI, etc., still fits into this step

**Data Preparation**

• Wrangle data and prepare it for training

• Clean that which may require it (remove duplicates, correct errors, deal with missing values, normalization, data type conversions, etc.)

• Randomize data, which erases the effects of the particular order in which we collected and/or otherwise prepared our data

• Visualize data to help detect relevant relationships between variables or class imbalances (bias alert!), or perform other exploratory analysis

• Split into training and evaluation sets .

**Data Compression**

When you work with large amounts of data, it becomes harder to come up with reliable solutions. Data reduction can be used to reduce the amount of data and decrease the costs of analysis.

After loading 23 the data, we separated the data into X and y where X is the image, and y is the label corresponding to X. The first layer/input layer for our model is convolution.

Convolution takes each pixel as a neuron, so we need to reshape the images such that each pixel value is in its own space, thus converting a 28x28 matrix of greyscale values into 28x28x1 tensor.

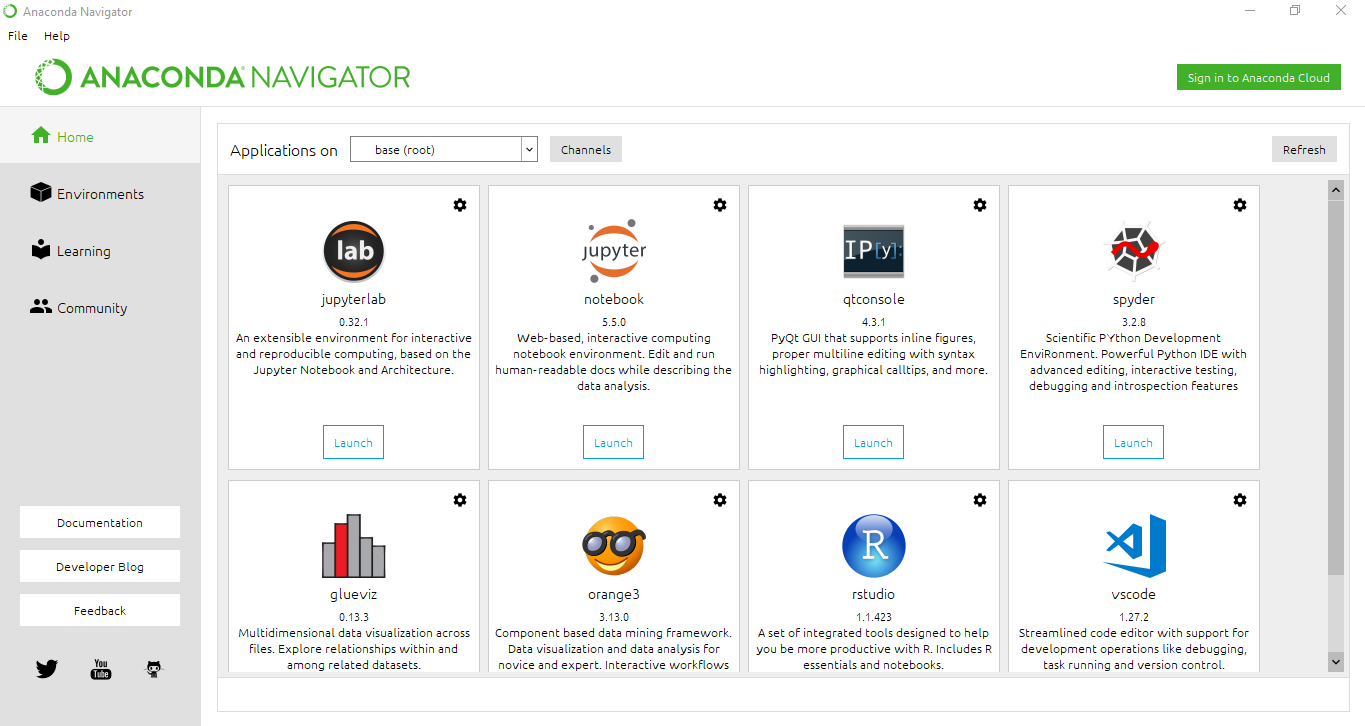
With the right dimensions for all the images, we can split the images into train and test for further steps. After loading the data, we separated the data into X and y where X is the image, and y is the label corresponding to X. The first layer/input layer for our model is convolution.

Convolution takes each pixel as a neuron, so we need to reshape the images such that each pixel value is in its own space, thus converting a 28x28 matrix of greyscale values into 28x28x1 tensor. With the right dimensions for all the images, we can split the images into train and test for further steps.

Model Description

Now, comes the fun part where we finally get to use the meticulously prepared data for model building. Depending on the data type (qualitative or quantitative) of the target variable (commonly referred to as the Y variable) we are either going to be building a classification (if Y is qualitative) or regression (if Y is quantitative) model.

Anaconda navigator



Anaconda Navigator is a desktop graphical user interface (GUI) included in Anaconda® distribution that allows you to launch applications and easily manage conda packages, environments, and channels without using command-line commands. Navigator can search for packages on Anaconda.org or in a local Anaconda Repository. It is available for Windows, macOS, and Linux.

Jupyter notebook



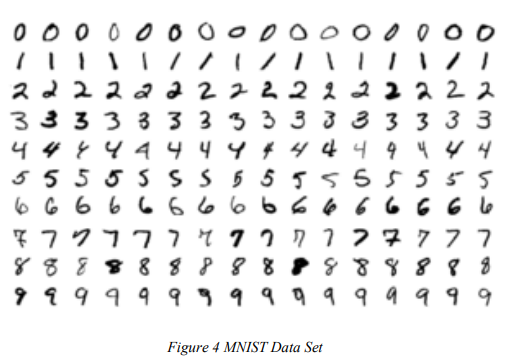
The Jupyter Notebook is an open source web application that you can use to create and share documents that contain live code, equations, visualizations, and text. Jupyter Notebook is maintained by the people at [Project Jupyter](http://jupyter.org/).

Jupyter Notebooks are a spin-off project from the IPython project, which used to have an IPython Notebook project itself. The name, Jupyter, comes from the core supported programming languages that it supports: Julia, Python, and R. Jupyter ships with the IPython kernel, which allows you to write your programs in Python, but there are currently over 100 other kernels that you can also use.

Training data & Test Data

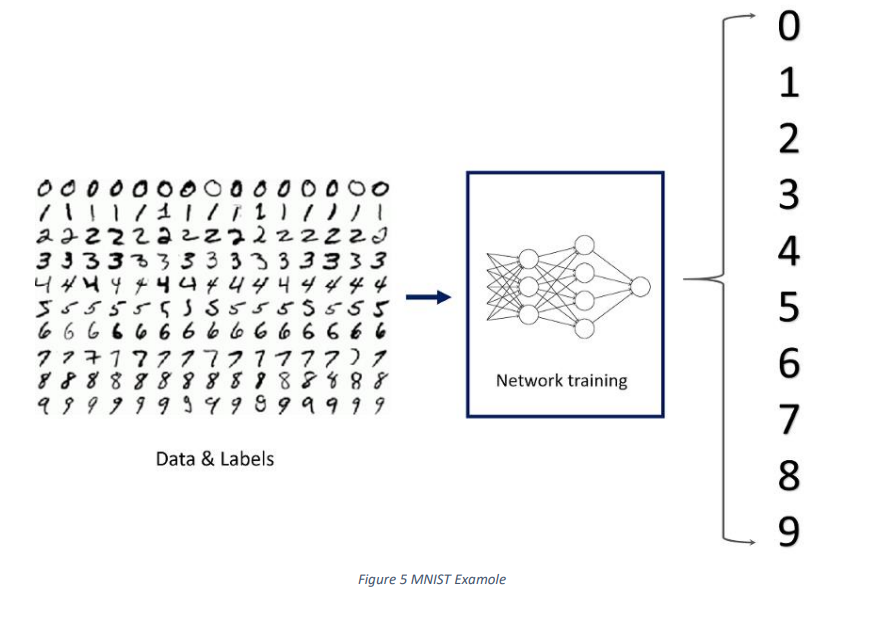
Modified National Institute of Standards and Technology (MNIST) is a large set of computer vision dataset which is extensively used for training and testing different systems. It was created from the two special datasets of National Institute of Standards and Technology (NIST) which holds binary images of handwritten digits.

The training set contains handwritten digits from 250 people, among them 50% training dataset was employees from the Census Bureau and the rest of it was from high school students. However, it is often attributed as the first datasets among other datasets to prove the effectiveness of the neural networks.



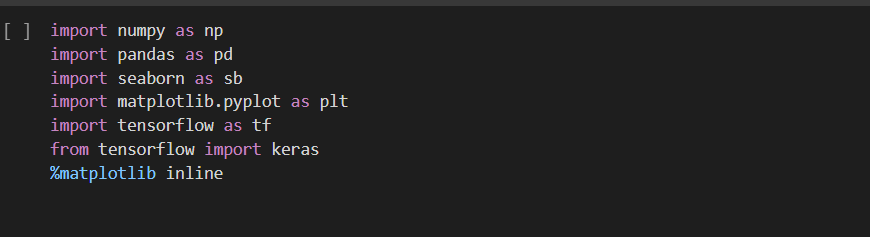
The database contains 60,000 images used for training as well as few of them can be used for crossvalidation purposes and 10,000 images used for testing. All the digits are grayscale and positioned in a fixed size where the intensity lies at the center of the image with 28×28 pixels.

Since all the images are 28×28 pixels, it forms an array which can be flattened into 28\*28=784 dimensional vector. Each component of the vector is a binary value which describes the intensity of the pixel.



Coding

Library inclusions

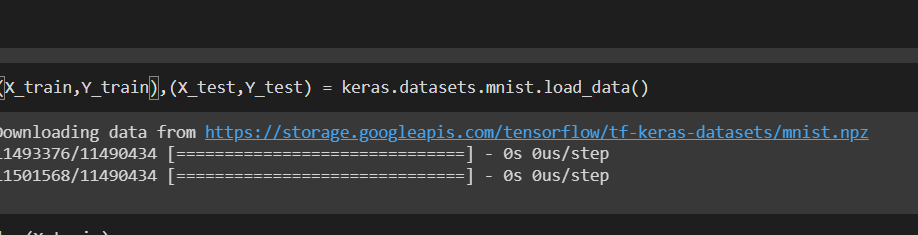


Numpy:-for numerical calculations regarding features.

Matplotlib:-for data visualization.

Tensorflow:-for constructing neural network.

Data set inclusion



All the data that is required to train model is shown above with its some top rows, data has been included in form of Images which has been already provided by the keras

Data mining & Data Science

**Data Exploration**

Data exploration is the first step in data analysis and typically involves summarizing the main characteristics of a data set, including its size, accuracy, initial patterns in the data and other attributes. It is commonly conducted by data analysts using visual analytics tools, but it can also be done in more advanced statistical software, Python**.** Before it can conduct analysis on data collected by multiple data sources and stored in data warehouses, an organization must know how many cases are in a data set, what variables are included, how many missing values there are and what general hypotheses the data is likely to support. An initial exploration of the data set can help answer these questions by familiarizing analysts with the data with which they are working.



**Figure 11:** Data Visualization

Machine learning algorithms

Machine learning algorithms could be broadly categorised to one of three types:

1. **Supervised learning** — In supervised learning, each example is a pair consisting of an input object (typically a vector) and a desired output value (also called the supervisory signal). A supervised learning algorithm analyzes the training data and produces an inferred function, which can be used for mapping new examples. An optimal scenario will allow for the algorithm to correctly determine the class labels for unseen instances. It is a machine learning task that establishes the mathematical relationship between input X and output Y variables. Such X, Y pair constitutes the labeled data that are used for model building in an effort to learn how to predict the output from the input. Supervised learning problems can be further grouped into regression and classification problems.

• Classification: A classification problem is when the output variable is a category, such as “red” or “blue” or “disease” and “no disease”.

• **Regression:** A regression problem is when the output variable is a real value, such as “dollars” or “weight”.

1. **Unsupervised learning** — is a machine learning task that makes use of only the input X variables. Such X variables are unlabeled data that the learning algorithm uses in modeling the inherent structure of the data. Unsupervised learning problems can be further grouped into clustering and association problems.

• Clustering: A clustering problem is where you want to discover the inherent groupings in the data, such as grouping customers by purchasing behavior.

• Association: An association rule learning problem is where you want to discover rules that describe large portions of your data, such as people that buy X also tend to buy Y.

**3. Reinforcement learning** — Reinforcement learning is an area of Machine Learning. It is about taking suitable action to maximize reward in a particular situation. It is employed by various software and machines to find the best possible behaviour or path it should take in a specific situation. Reinforcement learning differs from the supervised learning in a way that in supervised learning the training data has the answer key with it so the model is trained with the correct answer itself whereas in reinforcement learning, there is no answer but the reinforcement agent decides what to do to perform the given task. In the absence of a training dataset, it is bound to learn from its experience. It is a machine learning task that decides on the next course of action and it does this by learning through trial and error in an effort to maximize the reward.

• Input: The input should be an initial state from which the model will start

• Output: There are many possible output as there are variety of solution to a particular problem

• Training: The training is based upon the input, The model will return a state and the user will decide to reward or punish the model based on its output.

• The model keeps continues to learn.

• The best solution is decided based on the maximum reward.

**MODELS THAT CAN BE USED FOR THE PROJECT**

**SUPPORT VECTOR MACHINE**:

Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning. The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.

SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine. SVM algorithm can be used for Face detection, image classification, text categorization, etc. SVM can be of two types:

Linear SVM: Linear SVM is used for linearly separable data, which means if a dataset can be classified into two classes by using a single straight line, then such data is termed as linearly separable data, and classifier is used called as Linear SVM classifier.

Non-linear SVM: Non-Linear SVM is used for non-linearly separated data, which means if a dataset cannot be classified by using a straight line, then such data is termed as non-linear data and classifier used is called as Non-linear SVM classifier.

Example: SVM can be understood with the example that we have used in the KNN classifier. Suppose we see a strange cat that also has some features of dogs, so if we want a model that can accurately identify whether it is a cat or dog, so such a model can be created by using the SVM algorithm.

We will first train our model with lots of images of cats and dogs so that it can learn about different features of cats and dogs, and then we test it with this strange creature. So as support vector creates a decision boundary between these two data (cat and dog) and choose extreme cases (support vectors), it will see the extreme case of cat and dog. On the basis of the support vectors, it will classify it as a cat.

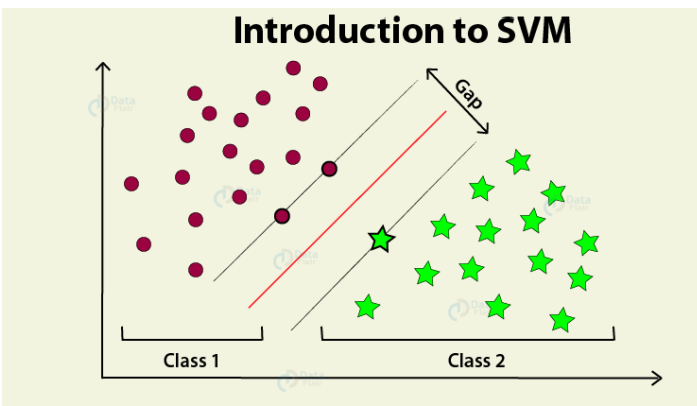


Figure: Working of a Support Vector Machine

The followings are important concepts in SVM –

**• Support Vectors** –

**Datapoints that are closest to the hyperplane is called support vectors**. Separating line will be defined with the help of these data points.

**• Hyperplane** − As we can see in the above diagram, it is a decision plane or space which is divided between a set of objects having different classes.

**• Margin** − It may be defined as the gap between two lines on the closet data points of different classes. It can be calculated as the perpendicular distance from the line to the support vectors. Large margin is considered as a good margin and small margin is considered as a bad margin. The main goal of SVM is to divide the datasets into classes to find a maximum marginal hyperplane (MMH) and it can be done in the following two steps –

• First, SVM will generate hyperplanes iteratively that segregates the classes in best way.

• Then, it will choose the hyperplane that separates the classes correctly. Pros of SVM classifiers

• SVM classifiers offers great accuracy and work well with high dimensional space. SVM classifiers basically use a subset of training points hence in result uses very less memory.

**Cons of SVM classifiers**

• They have high training time hence in practice not suitable for large datasets. Another disadvantage is that SVM classifiers do not work well with overlapping classes.

**2.K-NN ALGORITHM:**

K-Nearest Neighbour is one of the simplest Machine Learning algorithms based on Supervised Learning technique.

K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.

K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm. K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for the Classification problems.

K-NN is a non-parametric algorithm, which means it does not make any assumption on underlying data.

It is also called a lazy learner algorithm because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset. KNN algorithm at the training phase just stores the dataset and when it gets new data, then it classifies that data into a category that is much similar to the new data.

Example: Suppose, we have an image of a creature that looks similar to cat and dog, but we want to know either it is a cat or dog. So for this identification, we can use the KNN algorithm, as it works on a similarity measure. Our KNN model will find the similar features of the new data set to the cats and dogs images and based on the most similar features it will put it in either cat or dog category

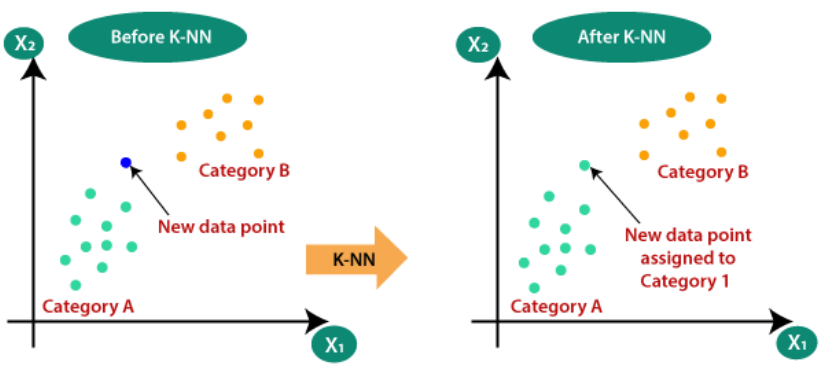


Figure to show Working of KNN

The K-NN working can be explained on the basis of the below algorithm:

Step-1: Select the number K of the neighbours

Step-2: Calculate the Euclidean distance of K number of Neighbours

Step-3: Take the K nearest Neighbours as per the calculated Euclidean distance.

Step-4: Among these k Neighbours, count the number of the data points in each category.

Step-5: Assign the new data points to that category for which the number of the neighbor is maximum.

Step-6: Our model is ready.

**Advantages of KNN Algorithm:**

It is simple to implement.

It is robust to the noisy training data.

It can be more effective if the training data is large.

**Disadvantages of KNN Algorithm:**

Always needs to determine the value of K which may be complex some time.

The computation cost is high because of calculating the distance between the data points for all the training samples.

Steps to implement the K-NN algorithm:

Data Pre-processing step o Fitting the K-NN algorithm to the Training set

Predicting the test result o Test accuracy of the result(Creation of Confusion matrix)

Visualizing the test set result.

This is pseudocode for implementing the KNN algorithm from scratch:

1. Load the training data.

2. Prepare data by scaling, missing value treatment, and dimensionality reduction as required.

3. Find the optimal value for K:

4. Predict a class value for new data:

1. Calculate distance (X, Xi) from i =1, 2, 3,….,n. where X= new data point, Xi= training data, distance as per your chosen distance metric.

2. Sort these distances in increasing order with corresponding train data.

3. From this sorted list, select the top ‘K’ rows.

• Find the most frequent class from these chosen ‘K’ rows. This will be your predicted class

After data encoding, the images and labels are ready to be fitted into our model. We need to define a Convolutional Neural Network Model.

**3.CONVOLUTION NEURAL NETWORK**:

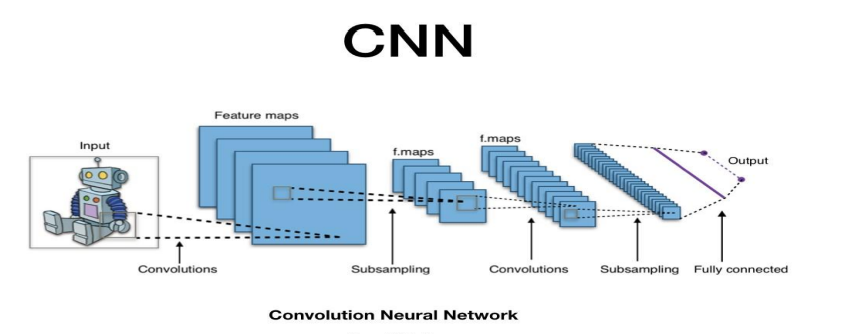
In simpler words, CNN is an artificial neural network that specializes in picking out or detect patterns and make sense of them. Thus, CNN has been most useful for image classification.

A CNN model has various types of filters of different sizes and numbers. These filters are essentially what helps us in detecting the pattern.

The convolutional neural network, or CNN for short, is a specialized type of neural network model designed for working with two-dimensional image data, although they can be used with one-dimensional and three-dimensional data. Central to the convolutional neural network is the convolutional layer that gives the network its name. This layer performs an operation called a “convolution“.

A CNN model generally consists of convolutional and pooling layers. It works better for data that are represented as grid structures, this is the reason why CNN works well for image classification problems. The dropout layer is used to deactivate some of the neurons and while training, it reduces offer fitting of the model.

Our model is composed of feature extraction with convolution and binary classification. Convolution and max pooling are carried out to extract the features in the image, and a 32 3x3 convolution filters are applied to a 28x28 image followed by a max-pooling layer of 2x2 pooling size followed by another convolution layer with 64 3x3 filters.



In the end, we obtain 7x7 images to flatten. Flatten layer will flatten the 7x7 images into a series of 128 values that will be mapped to a dense layer of 128 neurons that are connected to the categorical output layer of 10 neurons. The filter is smaller than the input data and the type of multiplication applied between a filter-sized patch of the input and the filter is a dot product.

A dot product is the element-wise multiplication between the filter-sized patch of the input and filter, which is then summed, always resulting in a single value. Because it results in a single value, the operation is often referred to as the “scalar product” Using a filter smaller than the input is intentional as it allows the same filter (set of weights) to be multiplied by the input array multiple times at different points on the input. Specifically, the filter is applied systematically to each overlapping part or filter-sized patch of the input data, left to right, top to bottom.

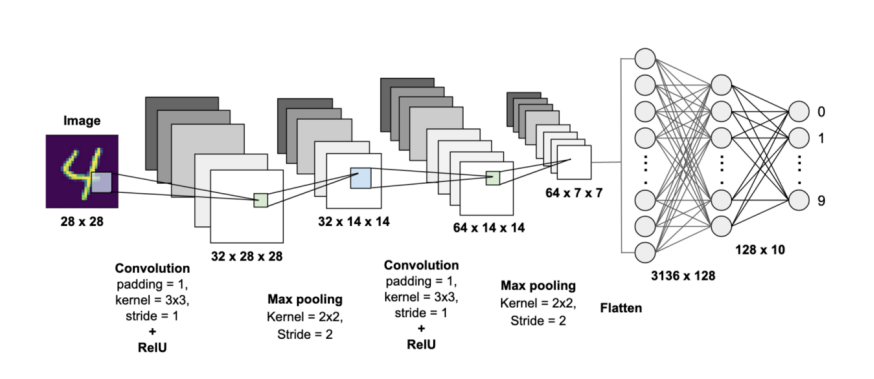
The output from multiplying the filter with the input array one time is a single value. As the filter is applied multiple times to the input array, the result is a two-dimensional array of output values that represent a filtering of the input. As such, the two-dimensional output array from this operation is called a “feature map”.

**WORKING OF CNN**:

Convolutional neural networks are composed of multiple layers of artificial neurons. Artificial neurons, a rough imitation of their biological counterparts, are mathematical functions that calculate the weighted sum of multiple inputs and outputs an activation value.

The behaviour of each neuron is defined by its weights. When fed with the pixel values, the artificial neurons of a CNN pick out various visual features.

When you input an image into a ConvNet, each of its layers generates several activation maps. Activation maps highlight the relevant features of the image. Each of the neurons takes a patch of pixels as input, multiplies their color values by its weights, sums them up, and runs them through the activation function.



Working : CNN for Hand Written Digit Recognition

The first (or bottom) layer of the CNN usually detects basic features such as horizontal, vertical, and diagonal edges. The output of the first layer is fed as input of the next layer, which extracts more complex features, such as corners and combinations of edges. As you move deeper into the convolutional neural network, the layers start detecting higher-level features such as objects, faces, and more. The operation of multiplying pixel values by weights and summing them is called “convolution” (hence the name convolutional neural network).

A CNN is usually composed of several convolution layers, but it also contains other components. The final layer of a CNN is a classification layer, which takes the output of the final convolution layer as input (remember, the higher convolution layers detect complex objects). Based on the activation map of the final convolution layer, the classification layer outputs a set of confidence scores (values between 0 and 1) that specify how likely the image is to belong to a “class.”

For instance, if you have a ConvNet that detects cats, dogs, and horses, the output of the final layer is the possibility that the input image contains any of those animals. After selecting the model the following process is done: The model type that we will be using is Sequential. Sequential is the easiest way to build a model in Keras. It allows you to build a model layer by layer.

We use the ‘add()’ function to add layers to our model. Our first 2 layers are Conv2D layers. These are convolution layers that will deal with our input images, which are seen as 2-dimensional matrices. 64 in the first layer and 32 in the second layer are the number of nodes in each layer. This number can be adjusted to be higher or lower, depending on the size of the dataset. In our case, 64 and 32 work well, so we will stick with this for now.

Kernel size is the size of the filter matrix for our convolution. So a kernel size of 3 means we will have a 3x3 filter matrix. Refer back to the introduction and the first image for a refresher on this. Activation is the activation function for the layer. The activation function we will be using for our first 2 layers is the ReLU, or Rectified Linear Activation. This activation function has been proven to work well in neural networks. 34 Our first layer also takes in an input shape.

This is the shape of each input image, 28,28,1 as seen earlier on, with the 1 signifying that the images are greyscale. In between the Conv2D layers and the dense layer, there is a ‘Flatten’ layer. Flatten serves as a connection between the convolution and dense layers. ‘Dense’ is the layer type we will use in for our output layer.

Dense is a standard layer type that is used in many cases for neural networks. We will have 10 nodes in our output layer, one for each possible outcome (0–9). The activation is ‘softmax’. Softmax makes the output sum up to 1 so the output can be interpreted as probabilities. The model will then make its prediction based on which option has the highest probability.

**Data cleaning**

Data cleaning is one of the important parts of machine learning. It plays a significant part in building a model. It surely isn’t the fanciest part of machine learning and at the same time, there aren’t any hidden tricks or secrets to uncover. However, proper data cleaning can make or break your project. Professional data scientists usually spend a very large portion of their time on this step. Because of the belief that, “Better data beats fancier algorithms”. If we have a well-cleaned dataset, we can get desired results even with a very simple algorithm, which can prove very beneficial at times. Obviously, different types of data will require different types of cleaning. However, this systematic approach can always serve as a good starting point.

**Training & Validation**

After the construction of the model the model has to be compiled to train it with the available data set. Optimizers are used to compile the model. Compiling the model takes three parameters: optimizer, loss and metrics. Optimizers are algorithms or methods used to change the attributes of the neural network such as weights and learning rate to reduce the losses. Optimizers are used to solve optimization problems by minimizing the function

The optimizer controls the learning rate. We will be using ‘adam’ as our optmizer. Adam is generally a good optimizer to use for many cases. The adam optimizer adjusts the learning rate throughout training. The learning rate determines how fast the optimal weights for the model are calculated. A smaller learning rate may lead to more accurate weights (up to a certain point), but the time it takes to compute the weights will be longer.

We will use ‘categorical\_crossentropy’ for our loss function. This is the most common choice for classification. A lower score indicates that the model is performing better. To make things even easier to interpret, we will use the ‘accuracy’ metric to see the accuracy score on the validation set when we train the model. The idea behind training and testing any data model is to achieve maximum learning rate and maximum validation. Better Learning rate and better validation can be achieved by increasing the train and test data respectively. Once the model is successfully assembled, then we can train the model with training data for 14 iterations, but as the number of iteration increases, there is a chance for overfitting. Therefore we limit the training up to 99% accuracy, as we are using real-world data for prediction, test data was used to validate the model.

**Different optimizers used in Neural Networks are**

1. Gradient Descent

2. Stochastic Gradient Descent (SGD)

3. Mini Batch Stochastic Gradient Descent (MB-SGD)

4. SGD with momentum

5. Nesterov Accelerated Gradient (NAG)

6. Adaptive Gradient (AdaGrad)

7. AdaDelta

8. RMSprop

9. Adam

**ADAM**

Optimizer Adaptive Moment Estimation is an algorithm for optimization technique for gradient descent. The method is really efficient when working with large problem involving a lot of data or parameters. It requires less memory and is efficient. Intuitively, it is a combination of the ‘gradient descent with momentum’ algorithm and the ‘RMSP’ algorithm.

The Adam optimization algorithm is an extension to stochastic gradient descent that has recently seen broader adoption for deep learning applications in computer vision and natural language processing. Adam is an optimization algorithm that can be used instead of the classical stochastic gradient descent procedure to update network weights iterative based in training data. Adam was presented by Diederik Kingma from OpenAI and Jimmy Ba from the University of Toronto in their 2015 ICLR paper (poster) titled “Adam: A Method for Stochastic Optimization“. The authors describe Adam as combining the advantages of two other extensions of stochastic gradient descent. Specifically:

• Adaptive Gradient Algorithm (AdaGrad) that maintains a per-parameter learning rate that improves performance on problems with sparse gradients (e.g. natural language and computer vision problems). Adaptive Moment Estimation is most popular today. ADAM computes adaptive learning rates for each parameter. In addition to storing an exponentially decaying average of past squared gradients vt like Adadelta and RMSprop, Adam also keeps an exponentially decaying average of past gradients mt, similar to momentum • Root Mean Square Propagation (RMSProp) that also maintains per-parameter learning rates that are adapted based on the average of recent magnitudes of the gradients for the weight (e.g. how quickly it is changing).

This means the algorithm does well on online and non-stationary problems (e.g. noisy). Properties of Adam:

1. Actual step size taken by the Adam in each iteration is approximately bounded the step size hyper-parameter. This property add intuitive understanding to previous unintuitive learning rate hyper-parameter.

2. Step size of Adam update rule is invariant to the magnitude of the gradient, which helps a lot when going through areas with tiny gradients (such as saddle points or ravines). In these areas SGD struggles to quickly navigate through them.

3. Adam was designed to combine the advantages of Adagrad, which works well with sparse gradients, and RMSprop, which works well in on-line settings. Having both of these enables us to use Adam for broader range of tasks. Adam can also be looked at as the combination of RMSprop and SGD with momentum.

**Why ADAM?**

1. Adam is an optimization algorithm that can be used instead of the classical stochastic gradient descent procedure to update network weights iterative based in training data.

2. Adam combines the best properties of the AdaGrad and RMSProp algorithms to provide an optimization algorithm that can handle sparse gradients on noisy problems.

3. Adam is relatively easy to configure where the default configuration parameters do well on most problems.

**Model Evaluation & Prediction**

For real-world image classification prediction, we need to do a little image pre-processing on the real-world images as model training was done with greyscale raster images. The steps of image pre-processing are :

1. Loading image

2. Convert the image to greyscale

3. Resize the image to 28x28

4. Converting the image into a matrix form

5. Reshape the matrix into 28x28x1 After pre processing, we predict the label of the image by passing the pre-processed image through the neural network. The output we get is a list of 10 activation values 0 to 9, respectively. The position having the highest value is the predicted label for the image.

These structures are called as Neural Networks. It teaches the computer to do what naturally comes to humans. Deep learning, there are several types of models such as the Artificial Neural Networks (ANN), Autoencoders, Recurrent Neural Networks (RNN) and Reinforcement Learning. But there has been one particular model that has contributed a lot in the field of computer vision and image analysis which is the Convolutional Neural Networks (CNN) or the ConvNet.

CNNs are a class of Deep Neural Networks that can recognize and classify particular features from images and are widely used for analyzing visual images. Their applications range from image and video recognition, image classification, medical image analysis, computer vision and natural language processing.

Methods for evaluating a model’s performance are divided into 2 categories: namely, holdout and Crossvalidation. Both methods use a test set (i.e data not seen by the model) to evaluate model performance. It’s not recommended to use the data we used to build the model to evaluate it. This is because our model will simply remember the whole training set, and will therefore always predict the correct label for any point in the training set. This is known as overfitting.

1. **Training set**

is a subset of the dataset used to build predictive models.

**2. Validation set** is a subset of the dataset used to assess the performance of the model built in the training phase. It provides a test platform for fine-tuning a model’s parameters and selecting the best performing model. Not all modeling algorithms need a validation set.

**3. Test set**, or unseen data, is a subset of the dataset used to assess the likely future performance of a model. If a model fits to the training set much better than it fits the test set, overfitting is probably the cause.

**Cross-Validation**:

Cross-validation is a technique that involves partitioning the original observation dataset into a training set, used to train the model, and an independent set used to evaluate the analysis. The most common cross-validation technique is k-fold cross-validation, where the original dataset is partitioned into k equal size subsamples, called folds. The k is a user-specified number, usually with 5 or 10 as its preferred value. This is repeated k times, such that each time, one of the k subsets is used as the test set/validation set and the other k-1 subsets are put together to form a training set. The error estimation is averaged over all k trials to get the total effectiveness of our model.

**CNN ARCHITECTURE**

CNNs are a class of Deep Neural Networks that can recognize and classify particular features from images and are widely used for analyzing visual images. Their applications range from image and video recognition, image classification, medical image analysis, computer vision and natural language processing. The term ‘Convolution” in CNN denotes the mathematical function of convolution which is a special kind of linear operation wherein two functions are multiplied to produce a third function which expresses how the shape of one function is modified by the other. In simple terms, two images which can be represented as matrices are multiplied to give an output that is used to extract features from the image. Technically, deep learning CNN models to train and test, each input image will pass it through a series of convolution layers with filters (Kernals), Pooling, fully connected layers (FC) and apply Softmax function to classify an object with probabilistic values between 0 and 1.

The below figure is a complete flow of CNN to process an input image and classifies the objects based on values. 6.1 Basic Architecture There are two main parts to a CNN architecture

• A convolution tool that separates and identifies the various features of the image for analysis in a process called as Feature Extraction

• A fully connected layer that utilizes the output from the convolution process and predicts the class of the image based on the features extracted in previous stages.

CNN Layers: The multiple occurring of these layers shows how deep our network is, and this formation is known as the deep neural network.

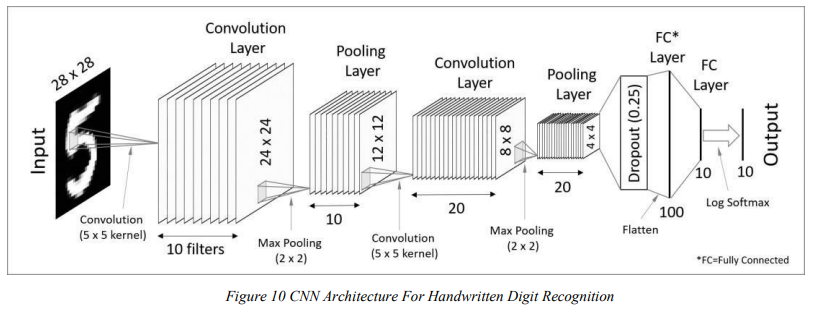
● Input: raw pixel values are provided as input.

● Convolutional layer: Input layers translates the results of the neuron layer. There is a need to specify the filter to be used. Each filter can only be a 5\*5 window that slides over input data and gets pixels with maximum intensities.

● Rectified linear unit [ReLU] layer: provided activation function on the data taken as an image. In the case of back propagation, ReLU function is used which prevents the values of pixels from changing.

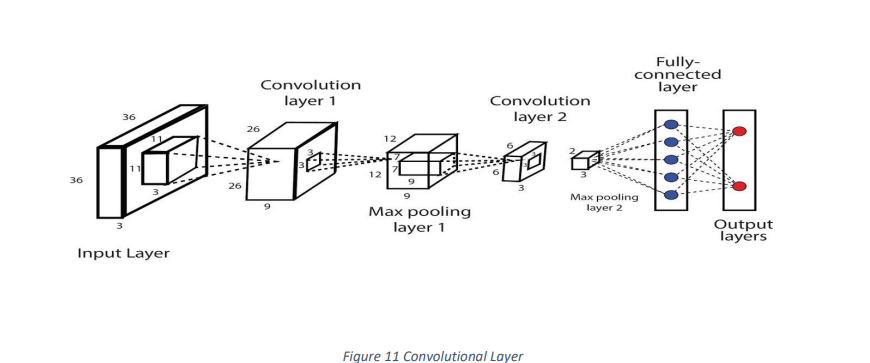
**● Pooling layer**: Performs a down-sampling operation in volume along the dimensions (width, height).

**Fully connected layer**: score class is focused, and a maximum score of the input digits is found. As we go deeper and deeper in the layers, the complexity is increased a lot. But it might be worth going as accuracy may increase but unfortunately, time consumption also increases.



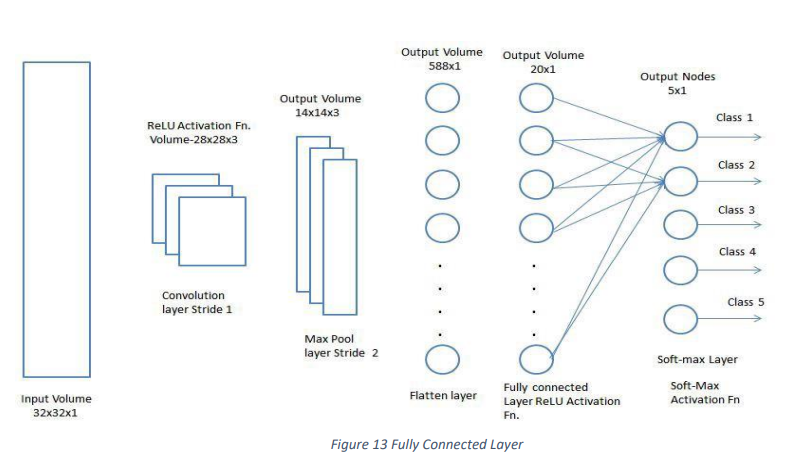
1. **Convolutional Layer**

This layer is the first layer that is used to extract the various features from the input images. In this layer, the mathematical operation of convolution is performed between the input image and a filter of a particular size MxM. By sliding the filter over the input image, the dot product is taken between the filter and the parts of the input image with respect to the size of the filter (MxM).



**Fully Connected Layer**

The Fully Connected (FC) layer consists of the weights and biases along with the neurons and is used to connect the neurons between two different layers. These layers are usually placed before the output layer and form the last few layers of a CNN Architecture.



**Activation Functions**

An activation function in a neural network defines how the weighted sum of the input is transformed into an output from a node or nodes in a layer of the network. Sometimes the activation function is called a “transfer function.” If the output range of the activation function is limited, then it may be called a “squashing function.” Many activation functions are nonlinear and may be referred to as the “nonlinearity” in the layer or the network design. The choice of activation function has a large impact on the capability and performance of the neural network, and different activation functions may be used in different parts of the model. Technically, the activation function is used within or after the internal processing of each node in the network, although networks are designed to use the same activation function for all nodes in a layer.

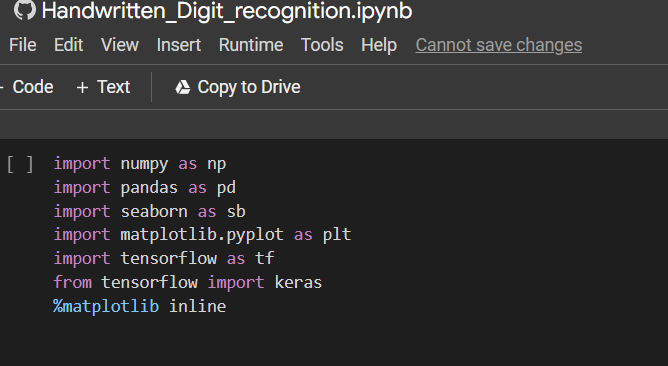
A network may have three types of layers: input layers that take raw input from the domain, hidden layers that take input from another layer and pass output to another layer, and output layers that make a prediction. All hidden layers typically use the same activation function. The output layer will typically use a different activation function from the hidden layers and is dependent upon the type of prediction required by the model.

Activation functions are also typically differentiable, meaning the first-order derivative can be calculated for a given input value.

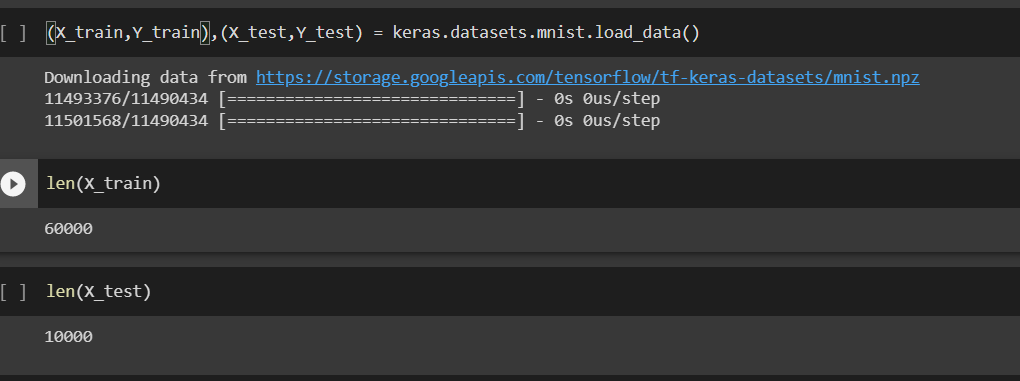
This is required given that neural networks are typically trained using the backpropagation of error algorithm that requires the derivative of prediction error in order to update the weights of the model. There are many different types of activation functions used in neural networks, although perhaps only a small number of functions used in practice for hidden and output layers. Finally, one of the most important parameters of the CNN model is the activation function. They are used to learn and approximate any kind of continuous and complex relationship between variables of the network. In simple words, it decides which information of the model should fire in the forward direction and which ones should not at the end of the network. It adds non-linearity to the network. There are several commonly used activation functions such as the ReLU, Softmax, tanH and the Sigmoid functions. Each of these functions have a specific usage. For a binary classification CNN model, sigmoid and softmax functions are preferred an for a multi-class classification, generally softmax us used.

Code Snippets Shots

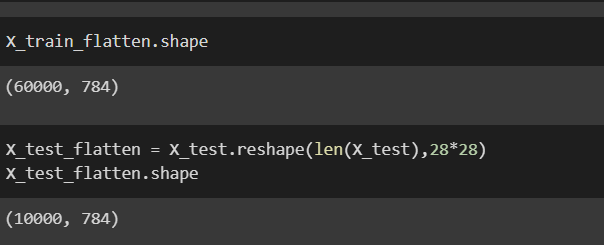
Importing Libraries



Train , Test Split



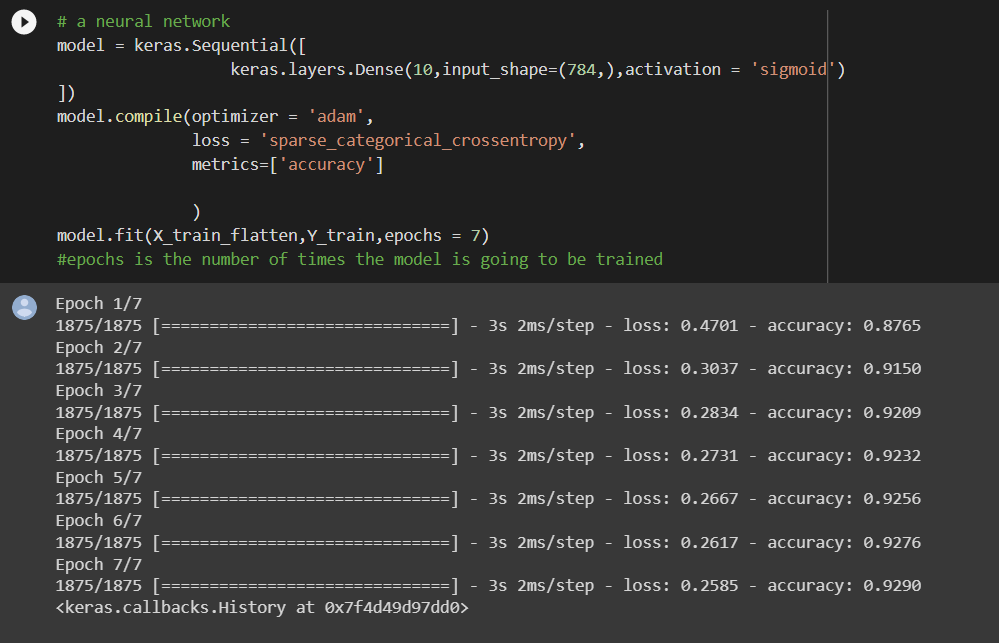
PreProcessing the Data



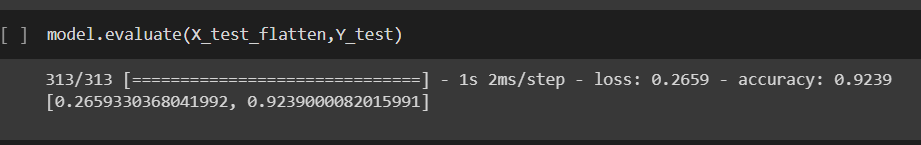
The image data cannot be fed directly into the model so we need to perform some operations and process the data to make it ready for our neural network. The dimension of the training data is (60000,28,28). The CNN model will require one more dimension so we reshape the matrix to shape (60000,28,28,1).

Create the model

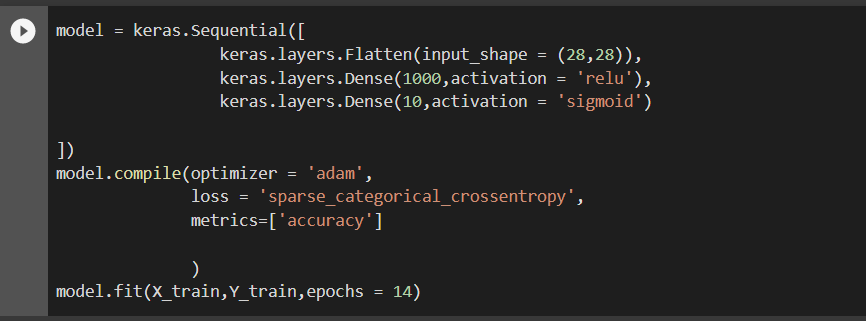
Now we will create our CNN model in Python data science project. A CNN model generally consists of convolutional and pooling layers. It works better for data that are represented as grid structures, this is the reason why CNN works well for image classification problems.



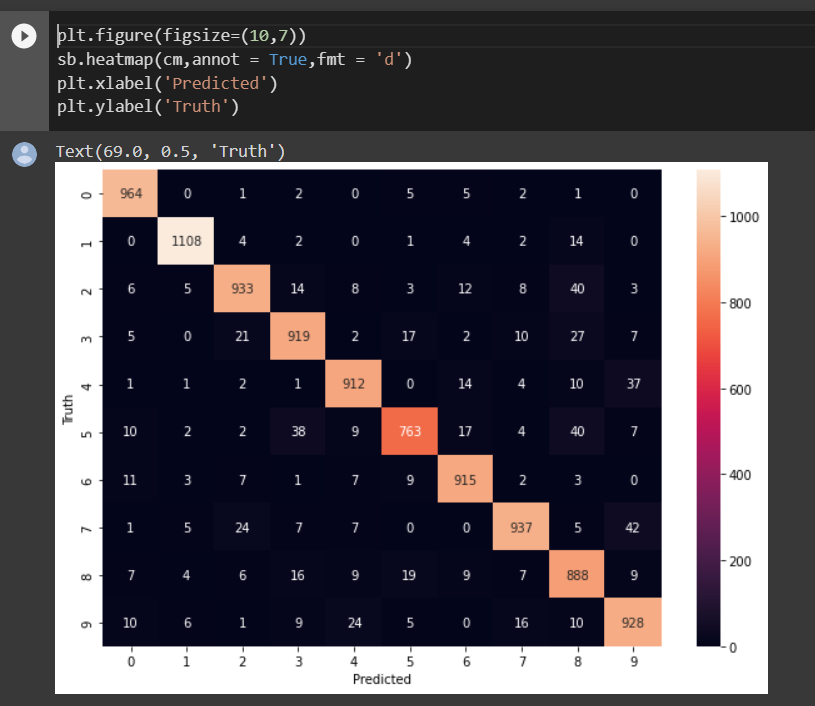
Evaluate the model



Improving the model



Figuring relation between truth and predicted points using Confusion Matrix



Conclusion

Our project HANDWRITTEN DIGIT RECOGNITION deals with identifying the digits.

The main purpose of this project is to build an automatic handwritten digit recognition method for the recognition of handwritten digit strings.

In this project, we applied CNN (Convolutional Neural Networks) architectures are used to achieve high performance on the digit string recognition problem

At last we achieved accuracy of 99%.

Future Scope

The proposed system takes 28x28 pixel sized images as input. The same system with further modifications and improvements in the dataset and the model can be used to build Handwritten Character Recognition System which recognizes human handwritten characters and predicts the output.

References

1. <https://www.tensorflow.org/learn>

2. <https://medium.com/the-andela-way/applying-machine-learning-to-recognize>-handwrittencharacters-babcd4b8d705

3. <https://nanonets.com/blog/handwritten-character-recognition/>

4. <https://www.geeksforgeeks.org/python-tkinter-tutorial/>

5. <https://medium.com/the-andela-way/applying-machine-learning-to-recognize> handwrittencharacters-babcd4b8d705

6. <https://nanonets.com/blog/handwritten-character-recognition/>

7. <https://www.tensorflow.org/learn>

8. [https://towardsdatascience.com/useful-plots-to-diagnose-your-neural-network 521907fa2f45](https://towardsdatascience.com/useful-plots-to-diagnose-your-neural-network%20521907fa2f45)

9. [https://towardsdatascience.com/the-best-machine-learning-algorithm-for-handwritten digitsrecognition-2c6089ad8f09](https://towardsdatascience.com/the-best-machine-learning-algorithm-for-handwritten%20digitsrecognition-2c6089ad8f09)

10. <https://www.analyticsvidhya.com/blog/2021/01/building-a-cnn-model-with-95-accuracy/>

11. [https://www.analyticsvidhya.com/blog/2020/10/what-is-the-convolutional-neural networkarchitecture/](https://www.analyticsvidhya.com/blog/2020/10/what-is-the-convolutional-neural%20networkarchitecture/)

Remarks

# REFERENCES