HandWritten Mathematical Equation Solver using CNN

A Project Report Submitted in the
Partial Fulfillment of the Requirements
for the Award of the Degree of

BACHELOR OF TECHNOLOGY

IN

INFORMATION TECHNOLOGY

Submitted by

Alavala Maneela	18881A1262
Kolepaka Pragnya	18881A1290
Nazeer Ahmad Abdul	18881A12A0

SUPERVISOR
Dr.Muni sekhar Velpuru
Associate Professor and Head

Department of Information Technology
VARDHAMAN COLLEGE OF ENGINEERING, HYDERABAD
An Autonomous Institute, Affiliated to JNTUH

April, 2022

VARDHAMAN COLLEGE OF ENGINEERING, HYDERABAD

An Autonomous Institute, Affiliated to JNTUH

Department of Information Technology

CERTIFICATE

This is to certify that the project titled **Hand Written Arithmetic Equation solver using CNN** is carried out by

Alavala Maneela	**18881A1262**
Kolepaka Pragnya	**18881A1290**
Nazeer Ahmad Abdul	**18881A12A0**

in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Information Technology** during the year 2021-22.

Signature of the Supervisor
Dr.Muni Sekhar Velpuru
Associate Professor and Head

Signature of the HOD
Dr.Muni Sekhar Velpuru
Associate Professor and Head

Project Viva-Voce held on _____

Examiner

Acknowledgement

The satisfaction that accompanies the successful completion of the task would be put incomplete without the mention of the people who made it possible, whose constant guidance and encouragement crown all the efforts with success.

We wish to express our deep sense of gratitude to **Dr.Muni Sekhar Velpuru**, Associate Professor and Project Supervisor, Department of Information Technology, Vardhaman College of Engineering, for his able guidance and useful suggestions, which helped us in completing the project in time.

We are particularly thankful to **Dr.Muni Sekhar Velpuru**, the Head of the Department, Department of Information Technology, his guidance, intense support and encouragement, which helped us to mould our project into a successful one.

We show gratitude to our honorable Principal **Dr. J.V.R. Ravindra**, for providing all facilities and support.

We avail this opportunity to express our deep sense of gratitude and heartful thanks to **Dr. Teegala Vijender Reddy**, Chairman and **Sri Teegala Upender Reddy**, Secretary of VCE, for providing a congenial atmosphere to complete this project successfully.

We also thank all the staff members of Information Technology department for their valuable support and generous advice. Finally thanks to all our friends and family members for their continuous support and enthusiastic help.

Alavala Maneela

Kolepaka Pragnya

Nazeer Ahmad Abdul

Abstract

Mathematics is important in our daily life. We use the calculators available in our electronic gadgets to solve mathematical equations. Calculators available in those gadgets are greatly advanced that they can even solve most complex equations. Developing a version that could recognize and remedy mathematical equations from handwriting is a great area of research.
This handwritten equation solver is a model that can solve mathematical equations written in different hand writings. Equations written by us is given as input to model. The model recognizes the digits, characters and symbols then forms a string of equation, solve it and gives it as output. Hand written math symbols data set is used to train the model. Output is generated using Convolutional Neural Network. We used Convolutional neural network to build model. Convolutional neural networks produce high accuracy in image processing. For classification purpose we use Convolutional Neural Network. After each and every character, symbol and digit is recognized they are appended to form a string and it is solved to generate roots which are given as output.

Keywords: Convolution neural Network; Classification

Table of Contents

List of Figures

Abbreviations

Abbreviation	**Description**
CNN	Convolutional Neural Network
OCR	Optimal Character Recognition
GLCM	Gray-level Co-Occurance Matrix
SVM	Support Vector Machine

CHAPTER 1

Introduction

1.1 Introduction

As the technology changes and advances, most of the written documents, books in the area of computing are also increasingly becoming digitalized. Mathematics is utilized in many areas. Digital document analysis and understanding is one of the major research concern today. For the recognition of English characters and numbers in electronic books OCR (optical character recognition) is mostly used to attain higher recognition accuracy. As the technology advances, machine learning and deep learning are increasingly becoming popular and also they started to play an important role in our day to day life . Now, machine learning and deep learning techniques are being employed in handwriting recognition, robotics, AI and lots of more fields.

Many fields, including education, engineering, and science, require handwritten mathematical formulas. The widespread availability of computationally capable touch-screen devices, analogous to the recent rise of deep neural networks as high-quality sequence recognition models, has led to broad acceptance of online handwritten mathematical expression recognition. Also, due to the global pandemic, a deeper study and improvement of such technologies is required to handle the current issues given by the widespread use of distance learning and remote work.Examples of applications for various disciplines and platforms demonstrate the usage of handwritten math recognition. A unique feature of the survey is that we looked at how different recognition algorithms are used in UI design, with the goal of assisting potential researchers in improving the performance of the introduced ways toward the best responses in actual applications.Finally, it gives an overview of future research areas in handwritten mathematical expression recognition and related applications.

Convolutional neural network (CNN) is mostly used classification model in computer vision area. Deep Convolutional Neural Network(CNN) has an outstanding performance in the field of image classification, machine learning and pattern recognition. CNN extract feature from images by series of operations. Here we mainly focused on the recognition of polynomial equation and after successful recognition of equation we apply character string operation for the solution of those equations. Feature extraction of each individual character is most difficult part for any handwritten character due to it's different shape and structure. For serving the purpose we apply CNN (Convolutional neural network) which did not require any predefined features for classification of characters. After character classification, recognition of character is done and then it solved.

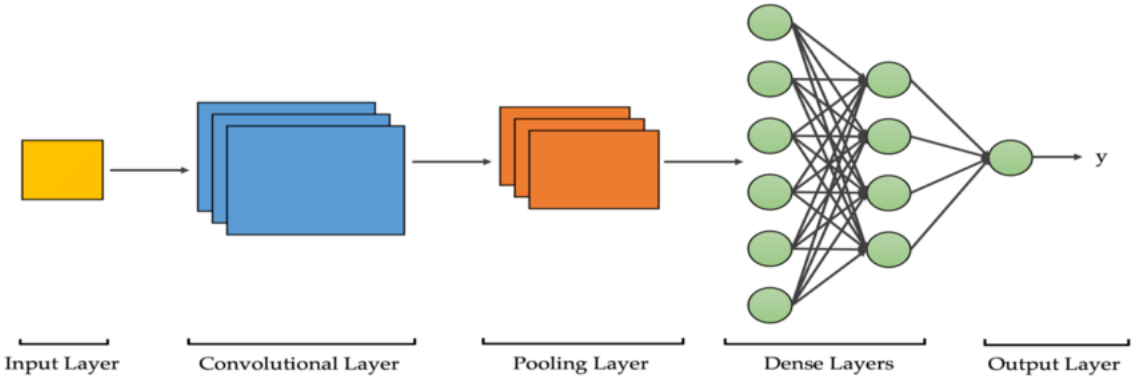

Figure 1.1: Overview of Convolutional Neural Network

1.2 Problem Statement

Calculators available in gadgets are so advanced that they can solve the equations, but we have to type those equations.Hand written mathematical equation solver is a system, that can recognize and solve equations by giving a handwritten equations as input. Handwritten mathematical expression recognition is still a most challenging job to do in the area of computer vision. The goal of this project is to create a model that will be able to recognize and determine the handwritten equation from its image by using the concepts of Convolution Neural Network and then the recognised equation is solved .

The major goal of the proposed system is understanding Convolutional Neural Network, and applying it to the handwritten recognition system, where it recognizes the digits and symbols, and then recognized equation is solved.

1.3 Objective of project

The main objective of our project is to recognise, solve and display the generated output of the given polynomial equation as input.Hand written mathematical equation solver uses handwritten math symbols dataset, from Kaggle, to train the model. After the model is trained, image of hand written equation is given as input. Contours are extracted from the image and then they are recognised.Then the recognised images are stored in string format. Then the recognised equation in the form of string is solved. The solution of that equation is given out as output. Convolutional Neural Network is used generate and train the model. The trained model then takes the input and generates output of the equation given as input.

1.4 Limitations of the project

Hand written mathematical equation solver model is limited to certain number of characters in the equation. As the number of symbols or digits increase in the equation, the accuracy of recognition decreases. The generated model takes image of hand written equation as input, generates contours to detect where digits are there in the given image. That is borders are generated for each digit or symbol in the image of equation.Therefore as digits or symbols increase in the equation, proper contours may not be generated, which ultimately leads to poor recognition. This decreases accuracy of the model.

This model is limited simple polynomial equations. The characters in the image should be bold for proper recognition. There should be some gap between two symbols, characters or digits for proper recognition. That is if two symbols, characters or digits overlap with each other they would be

considered as a single symbol, characters or digit and will be recognized as a class that matches the most according to the model. If any other symbol that is not trained, it will also be recognized to a class it matches the most. It does not throw any error, which is another limitation of our project.

1.5 Modules

Python is being used to create this project. We used jupyter notebook to construct the model. In this project we used Contour extraction to segment the symbols, digits and characters in the equation. Convolutional Neural Network is used to recognise the symbols, digits and characters in the equation.We used python programming language since it is easy to learn and work. The libraries we used include numpy, pandas, openCV, keras, pickle, matplotlib and others.

OpenCV is a huge open-source library for computer vision, machine learning, and image processing. We used openCV for contour extraction that is for segmentation of digits and symbols in the equation given as input. Tensorflow is used because it is an end-to-end open source platform for machine learning. It has a comprehensive, flexible ecosystem of tools, libraries and community resources that lets the researchers to push the state-of-art in machine learning, and developers easily build and deploy machine learning algorithms. Tensorflow consists of various libraries and keras is one of them.

Keras library is used in this project.Keras is an open-source high-level Neural Network library, which is written in Python and is capable enough to run on Theano, TensorFlow.It as an API designed for humans and it is highly flexible and easy to learn and use.In this project we used Keras library to design the neural network model.

Matplotlib library is used to view data in a particular required manner. We used marplotlib library in our project to view data similarities and differences. Pandas library is used to deal with dataframes, dataset and so on.

We used pandas library to read csv file. We initially convert image file to csv file. Later for easy operational purpose csv file is converted to dataframe. Again pandas is used to serve the purpose.To deal with arrays or whenever we required to create an array it is so easy to create by using numpy library. These are the libraries we used to construct the model.

Data preparation and model construction are the two modules of our project. Here data preparation include preparing data set. Converting data set to csv file. Removing any bias present in data set. All the classes present in data set does not contain equal number of images. Hence we considered equal number of images from all the classes. We hand removed the images that does not fit in the class.All these comes under data preparation. Model construction include constructing CNN using keras ans training the model using the data set we prepared through data preparation.

CHAPTER 2

Literature Survey

2.1 Literature Survey

So far, some work has been done on handwriting recognition. A system of math recognition MER based on SVM and a projection histogram of a simple math were proposed by [1] in . This is part of the offline recognition of handwritten expressions. This white paper basically focuses on a number of techniques used for feature extraction and recognition. An effective and robust system for recognizing printed and handwritten math characters has been proposed, using three consecutive paths: layout path and lexical path path and operator tree. Evaluate. The system takes math characters as input, and this input goes through three paths before the expression is evaluated [2]. Another system recognition system for offline handwritten mathematical symbols has been proposed. This feature extraction system considers the shape of the character . It is based on a relative study of feature extraction method . Detection was performed by an SVM support vector machine, a supervised machine learning algorithm that can be used for both classification and regression tasks.

Another system has been proposed that uses the diagonal feature extraction technique for handwritten characters using a feedforward neural network algorithm. This technique uses diagonal, horizontal, and vertical features for classification purposes. An article on math recognition using the convolutional neural network [3] is available online. Over the past years, a branch of artificial neural networks called Deep Learning has shown great potential in solving the problem of classification [4].

Around 2000 BC, the Babylonians solved a linear equation and Around

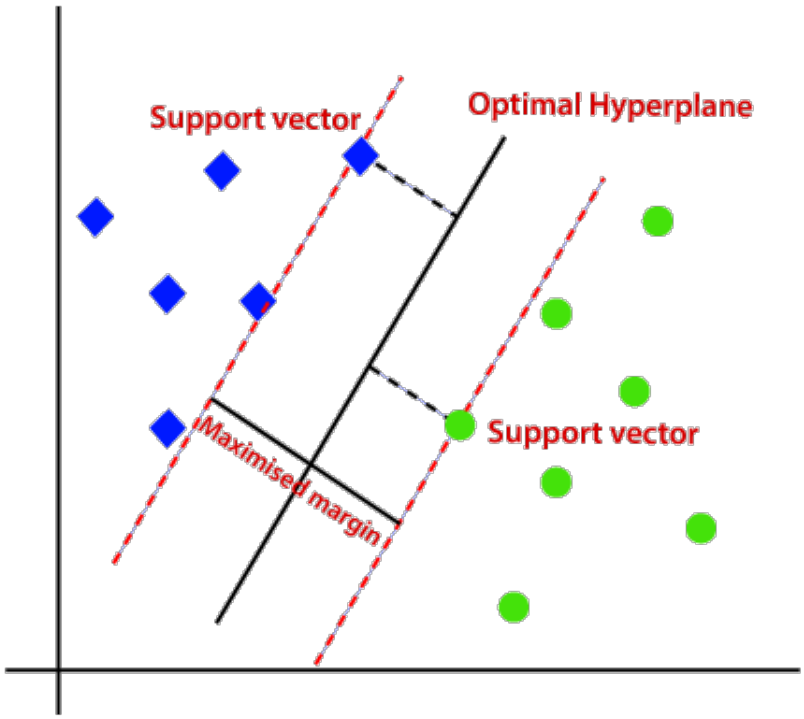

Figure 2.1: Support Vector Machine

300 BC BC Euclid developed a geometric approach to it Solves a quadratic equation in 1079 BC. Chr. Omar Khayyam (10501123) showed how to find the root of the cubic equation. Handwritten polynomial Detection and simplification Convolutional neural network 13th century Leonardo Fibonacci, 16th century Rene Descartes and Sir Isaac Newton play an important role in this Improve the polynomial. Many different papers can be found by hand Character segmentation . Some schemes also work Recognize the formula "MER". As "Using SVM and ME Projection Histogram Identification" . Some are for formulas printed offline And credits mentioned by Zanibbi et al. Al. (2002), "Recognizing printed mathematical symbols" , " SVM identification of mathematical symbols ", "Recognizing online math symbols using templates Consensus distance "," offline handwritten mathematics Symbol recognition by character geometry ". All things Suggested methods for recognizing symbols and Segmentation with various actions.

Recognition is Also performed by a CNN-based model of mathematical symbols And the letters . Discussion of stakeholders Labyrinth of online recognition of mathematical formulas . Most of these papers focused on cog-

nition Equation scheme. In this approach, the main node is on a handwritten polynomial Simplify the equation. Because there is no real job to do that. Handles handwritten image issues well. Or Previous works focused only on the level of recognition of It is an equation and does not focus on multivariable identification To the equation to solve the problem. This job motivates you to do so Handles multivariable segmentation and detection Efficiently display the results, each variable equation. After pretreatment, segmentation and recognition Input image, generate a string equation from this image, Simplify that the expression is the main goal. Much work is done in Almost three areas of math recognition Decades since the 1990s.

Chan et al. have Summarized and compared the various techniques available for Formula recognition. Mathematically Expression recognition can be divided into two main phases. Symbol recognition and structural analysis. symbol For segmentation, symbol segmentation and Recognition of segmented symbols. The former is Achieved in relation to various methods such as discovery Components, recursive horizontal and vertical projections Profile cut, recursive XY cut, progressive Grouping algorithm and the latter are performed in different ways Template matching, structure , Statistical approach like Chan et al. .. There are two main types of formula recognition. Online and offline. Previous sign recognition inside User input and the latter are later symbol recognition Entered by the user. Zanibbietal. produced promising results Use a tree transform to build a tree in Symbols in nodes to identify math formula. Garain et al. Utilize neuromotor function Handwriting features for symbol recognition Detailed feature extraction procedure. La Viola et al. has developed a system for creation and investigation Mathematical sketch created on a computer. Recent research in this area up to 2011 Zanibbietal. and mainly focused on online detection Of the formula sketched on the computer. Study in The field of mathematical sign recognition has prospered since then The emergence of deep learning. Mouchere et al. demonstrated Very promising results with online handwriting recognition Not just formulas Handwritten equations are still difficult structural patterns Recognition task.

2.2 Existing System

There are many existing projects on hand written character recognition.In the existing system the model used is different. The previous system utilized machine learning, image processing and classification based approaches to identify the symbols in the equation. In this system for feature extraction, shape of the character is considered. This system includes: Image preprocessing, segmentation of the characters in equation, feature extraction and classification [5] of symbol. Uses Gray-Level Co-Occurrence matrix (GLCM) for feature extraction and classification is done using support vector machine (SVM).Support Vector Machine which is a supervised machine learning algorithm that can be used for both classification or regression challenges. It detects the image and classify the images with the help of digital image processing.

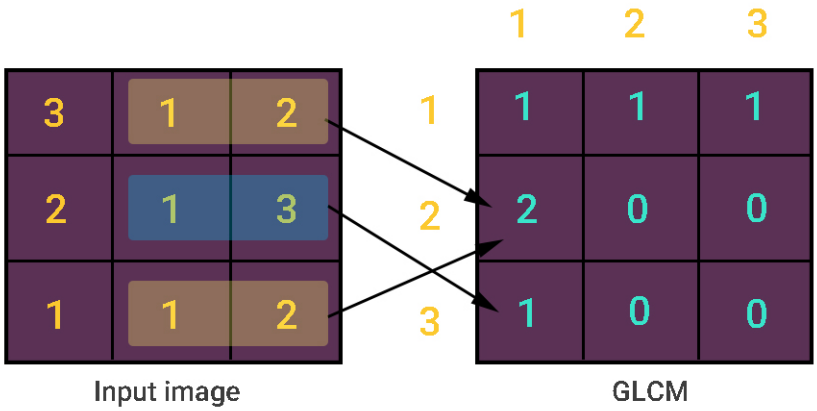

Figure 2.2: Gray-Level Co-Occurrence matrix

2.3 Limitations of Existing System

There were some limitations on existing systems.In existing system SVM classifier is made use to detect and recognise the symbols. But SVM algorithm is not suitable for large data sets. SVM does not perform very well when the data set has more noise i.e. target classes are overlapping. In cases where the number of features for each data point exceeds the number of training data samples, the SVM will underperform. But if we reduce number of images in

training data set, accuracy decreases. Hence less accuracy is one of the major drawback in existing system.

2.4 Proposed System

In our proposed method at first data set is processed.Each and every image is segmented using contour extraction. Then we consider each part of the segmented image as a full image for further process.Image is given as input to CNN model. Image is then preprocessed that is image is converted to gray scale and is resized. Preprocessing also includes removing bias. Initially data set is converted to csv file, where the csv file is of integer datatype. the csv file is further converted to dataframe for further usage purposes. Even for the classes such as symbols are assigned to some integers. In this project we have a total of 14 classes. The 14 classes are 10 digits, arithmetic symbols and x variable. Then for each symbol in equation image we then find specific characters in the form of connected component. Each segmented [6] character is then providing as input to the Convolutional Neural Network model for classification of the character. The resulting character that is the output of CNN is then used for making a character string which is similar to the original equation. Then the obtained equation string is evaluated and is projected as output. By using Convolutional Neural Network, accuracy is highly increased.Handwritten polynomial equation solver solves the given equation with high accuracy i.e; 99 percent and less computational time.

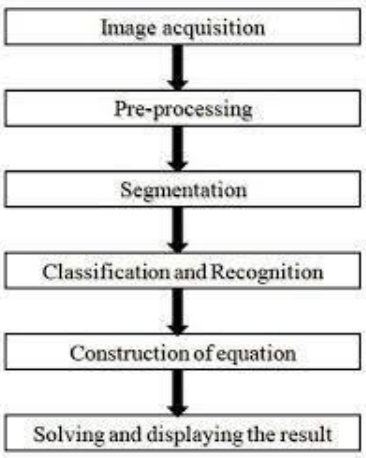

Figure 2.3: flow of proposed system

CHAPTER 3

Analysis

3.1 Introduction

Handwritten mathematical expression recognition is still one of the most challenging job in the area of computer vision. Due to the 2-D nesting assembly and their different sizes, the correction rate of symbol segmentation and recognition still cannot achieve its required requirements. The basic task for the recognition of mathematical expression is to segment the character or symbol or digit and then classify those character or symbol or digit. Image of a sequence of characters are segmented into sub images of individual symbols is Character segmentation. Contour extraction is done for character segmentation in this proposed model. This can be used at detection level. Firstly image is converted to gray scale then contours are generated. Contours are the borders of objects that is characters or symbols or digits. Examples for contour are given in figure 3.1 and figure 5.1.

Figure 3.1: Example of a contour

Figure 3.2: Example of a contour

The segmented characters are recognised. Various algorithms can be used for this purpose, but we used CNN due to its high accuracy. Recognition of digits, characters, symbols has been extremely found and studied. Various approaches in image processing and pattern recognition have been developed by scientists and engineers to solve this problem. Convolutional neural network

(CNN) is one of the mostly used classification model in computer vision area. CNN is one of the most popular models and has been providing the best recognition accuracy on object recognition. CNN extract feature from the image by a series of operations. Feature extraction of each individual character is most difficult part for any handwritten due to different shape and structure of the hand written character. To solve this problem we apply Convolutional neural network which does not require any predefined features for classification of specific character.

3.2 Software Requirement Specification

3.2.1 User Requirement

- Image of the equation

  Image of the equation that is written by us or image of the equation that is required to be solved is required as it is given as input to the model to generate required or necessary output.

- Desktop or laptop

  We implement the model or project in desktop or laptop. To give the input to model and to view the generated output we use laptop or desktop.

- Knowledge on using editor(Google colab or Jupyter notebook)

  Basic knowledge about editor is required to work on our project as we will be implementing our project in the editor we are comfortable with.

3.2.2 Software Requirement

- Windows, Linux, macbook and any platform which has x64 and x86 architecture can be used.

- Language : Python Programming Language

  Python is a high-level, interpreted programming language. Its design depicts the code readability with the use of significant indentation.

Python is dynamically-typed and garbage-collected. It is easy to use and easy to understand. Python is easy to work with as it contains huge library. Various machine learning and deep learning algorithms can be implemented using python with ease as it contains related required libraries.

- Editor: Google Collab or Jupyter Notebook

  Jupyter Lab is a web-based interactive development environment for Jupyter notebooks, code, and data. Jupyter Lab is flexible to configure and arrange the user interface to support a wide range of workflows in data science, scientific computing, and machine learning. Jupyter Lab is extensible and modular: write plugins that add new components and integrate with existing ones. We write the code, that is we design the model in Jupyter notebook.

- Library:Keras

  Keras is a powerful and easy-to-use free open-source Python library for developing and evaluating deep learning models. 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. Keras contains numerous implementations of commonly used neural-network building blocks such as layers, objectives, activation functions, optimizers, and a host of tools to make working with image and text data easier to simplify the coding necessary for writing deep neural network code. In addition to standard neural networks, Keras has support for convolutional and recurrent neural networks.

3.2.3 Hardware Requirement

- A camera or any smart phone Camera is used to take pictures or images of the hand written equation that is written by them.

- RAM: 4GB and above As the data set is large to run such huge model ram should be greater than equal to 4GB.

- Dataset-Handwritten math symbols dataset We require data set to train model. Without data set we cannot construct our project. Hence data set is one of the important requirements.
- Minimum of 1GB Hard Disk space for storage We store data set on the hard disk. The data set is nearly of size half GB. Hence we require a hard disk with minimum 1GB storage.
- The processor of core series i3,i5,i7.. with a computational speed of a minimum of 2.5 GHz

3.3 Flow chart and Algorithm

3.3.1 Algorithm

- Download the dataset, handwritten math symbols.
- Process each and every digit and also polynomial symbols.
- Extract contours from every image using contour extraction.In this way features are extracted from data.
- Build convolutional neural network model and train the data using convolutional neural network model.
- Test the model with images of equations.
- After taking image as input, segmentation of characters in equation is done .Segmentation is done using contour extraction.
- Each character in equation is recognised by the model and the equation is generated by appending the recognised characters as a string.
- The string equation is evaluated, considering the precedence.
- Generated output of equation is displayed to user.

3.3.2 Flow Diagram

Initially, client that is user sends image of equation to server that is hand written polynomial equation solver model. Then the model takes that image as input and perform Image Preprocessing which includes conversion image to gray scale and resizing, image segmentation that is contours of each symbol is generated, symbol classification, equation string formation and then generated equation is solved using python functions. Then server outputs the result generated to client that is user.

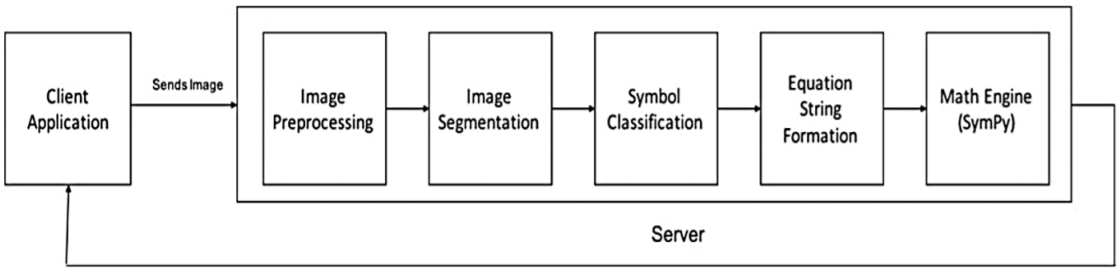

Figure 3.3: Flow Diagram

- Image Preprocessing includes removing any bias present in the dataset. It also includes converting data set into csv file. Csv file is of integer data type. Image processing includes all the pre work that is done to images to prepare them to train the model. Here same digits are assigned to digits and numbers from 10 to 14 are assigned for other symbols and x variable. These are few steps that include or contribute to image preprocessing.

- Image segmentation includes segmenting the images into sub images where sub images are the individual characters or symbol or digits. All the symbols or digits present in the image are segmented or separated into contours. After the contours are extracted bounding rectangles are formed. Where bounding rectangles specify the position of contour in the image. We used contour extraction for image segmentation. In python opencv is used for contour extraction. contours are the lines that borders a object where in our case objects are digits or symbols or x variable.

- After bounding rectangles are formed each and every contour are classified using constructed CNN model. Symbol classification includes construction and training of CNN model so as to be used in symbol classification. We trained the model for 14 classes. 14 classes include 10 digits, arithmetic operators and x variable. After construction each and every segmented sub image is classified to one of the 14 classes.

- After classification each and every classified symbol is concatenated to form a string. Initially, empty string is considered and each and every classified digit or symbol or x variable is appended to string respectively.This generates the equation in string format.

- After string is formed, it is again converted into list format. Digits are converted to numbers accordingly. Highest degree of the equation is identified. Then an empty list of size highest degree is considered or generated. Co-efficients of each power as identified are placed in their position the list considered.Then we use python library to generate roots. Generated roots are given as output.

CHAPTER 4

Design

4.1 Introduction

In design process we examine more than 1,00,000 photos of digits and symbols that have been assigned 14 different class labels. Each class label represents a digit or symbol. There are total of ten digits from zero to nine and also arithmetic symbols such as plus '+', minus '-', multiply 'x', divide '/' and also variable 'x' are considered. Each class is labeled with an integer. Class labels for digits are digits itself. For symbols some integer is assigned as class label. The photos are resized and predictions are done on these down scaled images. The user gives the image of equation as input to the model that takes the image and convert it to gray scale image and the gray scale image will be segmented. During this phase, we plan how the model will be implemented. Segmentation is done using contour extraction. Here we use openCV to implement contour extraction.

4.2 UML Diagrams

A UML diagram is a diagram based on the UML (Unified Modeling Language) with the purpose of visually representing a system along with its main actors, roles, actions, artifacts or classes, in order to better understand, alter, maintain, or document information about the system.

4.2.1 Usecase Diagram

In UML, use-case diagrams model the behavior of a system and help to capture the requirements of the system. Use-case diagrams describe the high-level functions and scope of a system. These diagrams also identify the interactions between the system and its actors.Here the actors are client that

is user and system that is server.There are various use cases such as feature extract,input image etc.

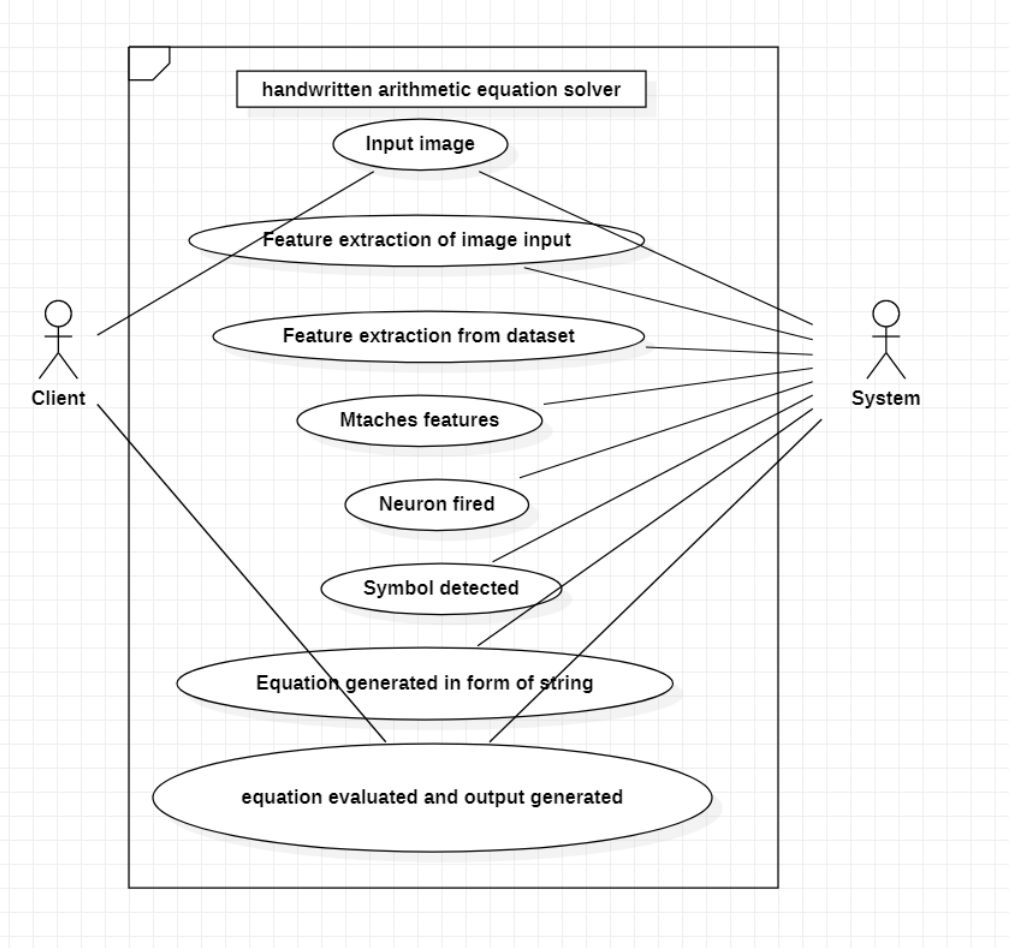

Figure 4.1: Usecase Diagram-Handwritten polynomial equation solver

User gives hand written equation as input and will only be able to access the output obtained from the model.Here User and system(server) are actors. System(server) involves with all the functions that are input image, feature extraction from input image, feature extraction from data set, matches features, neuron fired, symbol detected, equation generated in form of string, equation evaluated and output generated. Both actors have scope in input image and equation evaluated and output generated.

4.2.2 Flow Chart

A flowchart is a type of diagram that represents a workflow or process.Flowcharts are used in analyzing, designing,documenting or managing a

process or program in various fields.A Flowchart is a graphically representation of the structure of process or system, algorithm or the step-by-step solution of the problem. The Flowchart describes the flow of data through an information processing systems and the parts of the flows. The flow is a set of the logic operations that meet the certain requirements.

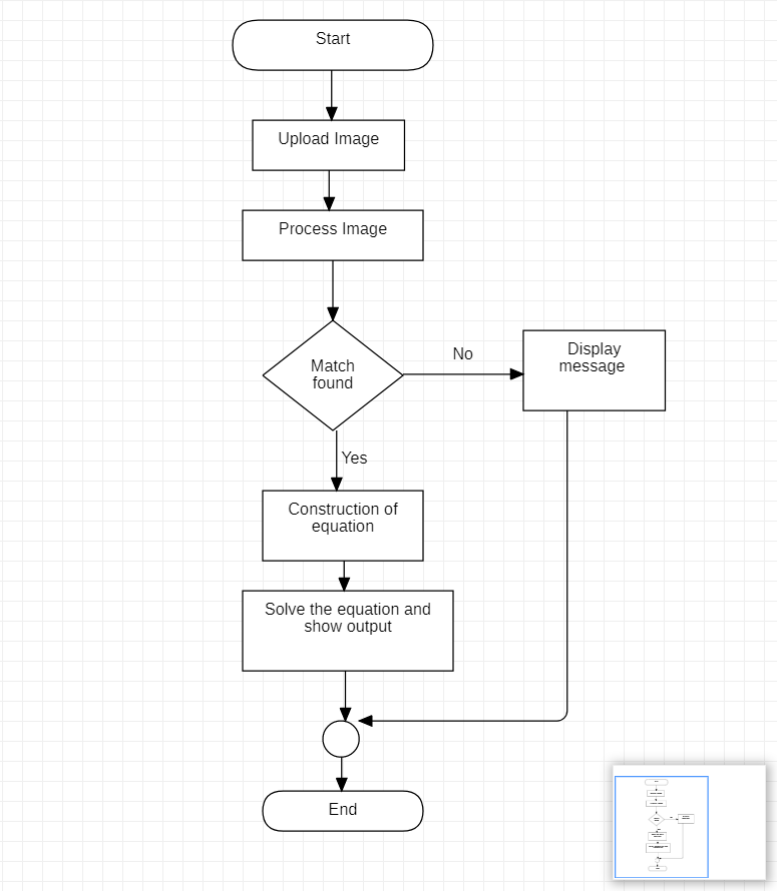

Figure 4.2: Flow analysis of the model

After the user gives hand written equation image as input to model, image processing is done, then classification of image is done. From all the 14 classes the symbol that mostly matches with input symbol is appended to equation string. After the equation string is formed it is solved to output the roots of the polynomial equation and solution of any arithmetic equation.

4.3 Module Design and Organization

The data is gathered from the handwritten math symbols data set.The data set consists many math symbols and all digits from which we only consider arithmetic symbols, all digits and x variable which are required. As to remove bias present in data set if symbol has significantly more number of images than the other symbol it creates bias in the data set. Hence we consider equal number of images from each symbol. Here 5000 images from each symbol is considered to train the model. Equal number of images from each symbol is considered. Also images that does not fit a particular class are also removed but checking all the images in every class. Then data set is converted to csv file. Csv file is of integer data typr. For digits, digits itself are assigned to represent them. As for symbols and x variable number from 10 to 13 are used. Hence integer csv file is generated. Then it is converted to dataframe accordingly to serve the further purposes.

Convolutional nueral network is constructed. In the project we used keras to generate convolution neural network. The generated csv file is given to model to train the model and the model is trained in such a way that it recognises the features of every symbol.Every symbol has it's own characteristics,With the help of those distinct characteristics the machine is able to predict the features or patterns. We train the CNN model. Since we used CNN the pattern recognition is good and the accuracy will be high.Total of thirty epochs are performed which took nearly three hours for processing. The training accuracy and validation accuracy increased for every epoch and the training loss and validation loss is decreased for every epoch. After increasing epochs more than 30, change in accuracy and loss is less, but time taken to process the model is increasing. Hence only 30 epochs are performed.

CHAPTER 5

Implementation

5.1 Introduction

The Proposed system, we implemented is Handwritten Mathematical Equation solver using CNN. This project helps us to calculate the arithmetic equations, polynomial equations easily and accurately. The accuracy of this project is 98 percent approximately. The main advantage of this project is it reduces human effort and makes calculation easily. Typing equations can be tiresome. Also the scientific calculators that are available can solve polynomial equations only up to degree 4. But the proposed project can be used to solve equations up to the degree of n. Here we demonstrate a model which can recognize handwritten arithmetic equation, solve that equation and also polynomial equations and generate roots of all the polynomial equations with degree up to n.

First after taking image as input individual characters, symbols or digits present in image are located in image as bounding rectangles. Contours are formed from them. Then recognition of individual character, symbol or digit takes place. As seen from the results of the experiment, CNN proves to be far better than other classifiers. A convolutional neural network is a deep learning algorithm that takes an input image, assigns importance to different objects in the image, and distinguishes them from each other. The results can be made more accurate with more convolution layers and a greater number of hidden neurons. It can completely abolish the need for typing. In future we will try to improve the accuracy and try to make the system workable for multiple mathematical formula simultaneously.

5.2 Technology

Hand-Written mathematical Equation solver using CNN(Convolutional Neural Network) is very useful for solving mathematical equations.It helps to solve mathematical equations accurately .It is widely used in applications of image and video recognition and also in recommend systems and Natural Language Processing(NLP). However, convolution is more effective because it decreases the number of parameters/objects which makes different from other deep learning models. CNN is a Deep Learning Algorithm which takes image as input and learns various features of image through filters and counters.It is most popular classifier for image processing and it is popularly used in analyzing images. It takes an input image, assigns importance to different objects in the image, and distinguishes them from each other.This allows them to learn about the important objects present in the image that allows them to discern one image from the other.It receives the input and transforms the input into some output and this output is transformed to next layer as input.

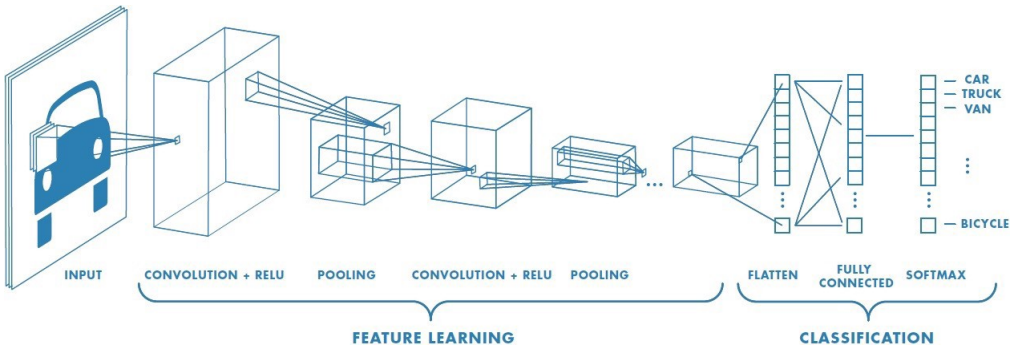

Figure 5.1: Convolutional Neural Network

5.3 Method of Implementation

Implementation of hand written equation solver model is done using Convolutional neural network. Initially, Hand written math symbols data set is taken from kaggle which contains over 100000 images.In the data set some symbols have more number of images than others. To remove this bias consider same number of images from all the digits and symbols. In this model we only

consider 14 different classes that is 0 to 9 digits,+,-,multiplication symbol and x variable.

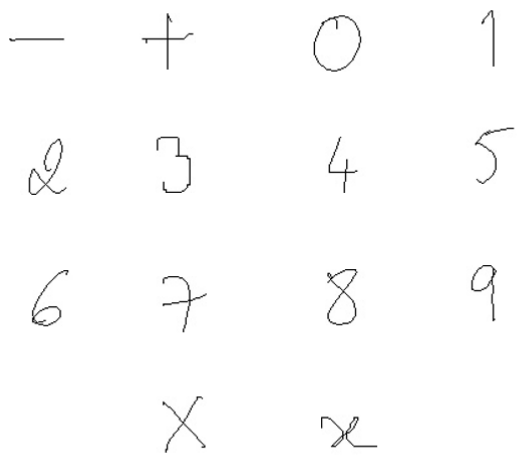

Figure 5.2: Sample images from data set

We trained the model using the data from data set where we considered same number of images for each digit or symbol to remove bias. Initially 500 images from each symbol is considered. Time taken to process all the training images is nearly 10 minutes but the accuracy achieved is less. We first run the model for 30 epochs. After 30 epochs the loss is not considerably decreasing. The difference in accuracy and loss after 30 epochs is differing in only decimals. Hence we stopped at 30 epochs.

By considering more number of images training of model is done again to verify the results. We increased number of images from each symbol.Now we considered 1000 images from each symbol. Time taken to process all the training images is more than that of 500 images from each class but the accuracy achieved is also relatively more.

```
In [15]: data7=[]
data7=load_images_from_folder1("C:/Users/HP/Documents/hand written poly equation solver/img files/7")
for i in range(0,len(data7)):
    data7[i]=np.append(data7[i],['7'])
data=np.concatenate((data,data7))
print(len(data))

5500

In [16]: data8=[]
data8=load_images_from_folder1("C:/Users/HP/Documents/hand written poly equation solver/img files/8")

for i in range(0,len(data8)):
    data8[i]=np.append(data8[i],['8'])
data=np.concatenate((data,data8))
print(len(data))

6000

In [17]: data9=[]
data9=load_images_from_folder1("C:/Users/HP/Documents/hand written poly equation solver/img files/9")

for i in range(0,len(data9)):
    data9[i]=np.append(data9[i],['9'])
data=np.concatenate((data,data9))
print(len(data))

6500

In [18]: data13=[]
data13=load_images_from_folder1("C:/Users/HP/Documents/hand written poly equation solver/img files/times")

for i in range(0,len(data13)):
    data13[i]=np.append(data13[i],['13'])
data=np.concatenate((data,data13))
print(len(data))

7000
```

Figure 5.3: Data preparation of 500 images from each class

```
In [14]: model.compile(loss="categorical_crossentropy",optimizer='adam',metrics=['accuracy'])

In [15]: model.fit(np.array(l),cat,shuffle=True,epochs=30)

Epoch 1/10
219/219 [==============================] - 5s 22ms/step - loss: 2.2267 - accuracy: 0.4450
Epoch 2/10
219/219 [==============================] - 5s 22ms/step - loss: 0.7095 - accuracy: 0.7873
Epoch 3/10
219/219 [==============================] - 5s 21ms/step - loss: 0.4324 - accuracy: 0.8674
Epoch 4/10
219/219 [==============================] - 5s 21ms/step - loss: 0.3175 - accuracy: 0.9024
Epoch 5/10
219/219 [==============================] - 4s 20ms/step - loss: 0.2628 - accuracy: 0.9200
Epoch 6/10
219/219 [==============================] - 4s 19ms/step - loss: 0.2016 - accuracy: 0.9359
Epoch 7/10
219/219 [==============================] - 4s 19ms/step - loss: 0.1676 - accuracy: 0.9453
Epoch 8/10
219/219 [==============================] - 4s 19ms/step - loss: 0.1309 - accuracy: 0.9569
Epoch 9/10
219/219 [==============================] - 4s 19ms/step - loss: 0.1293 - accuracy: 0.9589
Epoch 10/10
219/219 [==============================] - 4s 19ms/step - loss: 0.1026 - accuracy: 0.9659

Out[15]: <tensorflow.python.keras.callbacks.History at 0x2106c0c6340>
```

Figure 5.4: Accuracy achieved by considering 500 images from each class

5.3.1 Construction of Convolution Neural Network

Same process is repeated by increasing the number of images. Considering 1500, 2000, 2500 and 5000 images from each class, model is trained. The results that is accuracy is increasing as the number of images is increasing and also loss is decreased. But the time taken in processing data increased significantly as the number of images increased. Finally, model is trained by considering 5000 images from each symbol. Accuracy achieved by considering 5000 images from each class is nearly 98 percent. We trained the model for

```
In [15]: data7=[]
data7=load_images_from_folder1("C:/Users/HP/Documents/hand written poly equation solver/img files/7")
for i in range(0,len(data7)):
    data7[i]=np.append(data7[i],['7'])
data=np.concatenate((data,data7))
print(len(data))

11000

In [16]: data8=[]
data8=load_images_from_folder1("C:/Users/HP/Documents/hand written poly equation solver/img files/8")

for i in range(0,len(data8)):
    data8[i]=np.append(data8[i],['8'])
data=np.concatenate((data,data8))
print(len(data))

12000

In [17]: data9=[]
data9=load_images_from_folder1("C:/Users/HP/Documents/hand written poly equation solver/img files/9")

for i in range(0,len(data9)):
    data9[i]=np.append(data9[i],['9'])
data=np.concatenate((data,data9))
print(len(data))

13000

In [18]: data13=[]
data13=load_images_from_folder1("C:/Users/HP/Documents/hand written poly equation solver/img files/times")

for i in range(0,len(data13)):
    data13[i]=np.append(data13[i],['13'])
data=np.concatenate((data,data13))
print(len(data))

14000
```

Figure 5.5: Data preparation of 1000 images from each class

```
In [14]: model.compile(loss="categorical_crossentropy",optimizer='adam',metrics=['accuracy'])

In [15]: model.fit(np.array(l),cat,shuffle=True,epochs=10)

Epoch 1/10
438/438 [==============================] - 5s 11ms/step - loss: 1.7971 - accuracy: 0.5371
Epoch 2/10
438/438 [==============================] - 5s 12ms/step - loss: 0.5341 - accuracy: 0.8387
Epoch 3/10
438/438 [==============================] - 5s 12ms/step - loss: 0.3254 - accuracy: 0.8996
Epoch 4/10
438/438 [==============================] - 5s 12ms/step - loss: 0.2459 - accuracy: 0.9235
Epoch 5/10
438/438 [==============================] - 5s 12ms/step - loss: 0.1929 - accuracy: 0.9379 0s - loss: 0.1928 - accura
Epoch 6/10
438/438 [==============================] - 5s 12ms/step - loss: 0.1617 - accuracy: 0.9490
Epoch 7/10
438/438 [==============================] - 6s 13ms/step - loss: 0.1520 - accuracy: 0.9515
Epoch 8/10
438/438 [==============================] - 6s 13ms/step - loss: 0.1246 - accuracy: 0.9607
Epoch 9/10
438/438 [==============================] - 6s 13ms/step - loss: 0.1233 - accuracy: 0.9606
Epoch 10/10
438/438 [==============================] - 5s 13ms/step - loss: 0.1043 - accuracy: 0.9662

Out[15]: <tensorflow.python.keras.callbacks.History at 0x22a7ef57b50>
```

Figure 5.6: Accuracy achieved by considering 1000 images from each class 30 epochs. Some symbols have even more images but some have less images. So as to remove the bias we only considered 5000 images from each symbol.

A CNN layer consists of many hidden layers called convolutional layers.The layer receives the input and transforms the input into some way and outputs the transformed input to next layer. Each layer consists of different kind of filters that detects the patterns. These layers consists of many types of filters such as edge detectors, corner detectors etc.The deeper the network goes more sophisticated the patterns becomes.

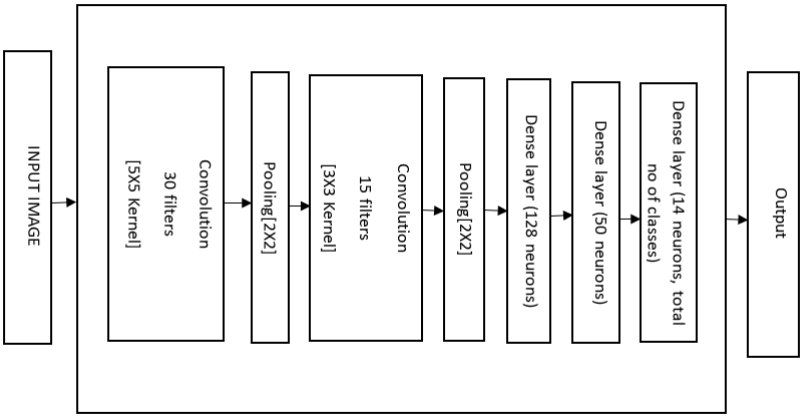

Figure 5.7: Convolutional neural network

At first the hand written image is given as input.Input image is given to the model by resizing the image after converting it into grey scale.

Initially, the image is given as input to convolution layer which consists of 30 filters with kernel size of size 5X5 by using ReLu activation function,ReLu(Rectified Linear Unit) is a linear function that will output the input itself, if input is positive, otherwise, it will output zero. Output from convolutional layer is given as input to Max Pooling Layer, with kernel of size 2X2. It is a pooling operation that selects the maximum element from the region of the feature map covered by the filter. Then the output from max pooling layer is given as input to Convolution layer which consists of 15 filters with of size 3X3 kernel. Then output from previous layer will be given to Max Pooling Layer of size 2X2. Finally, output from pooling layer is flattened and it passes 3 dense layers . in which the first one consists 128 neurons , second consists of 50 neurons and third consists 14 neurons(total number of classes). After passing through all these layers output is generated.

Important layers and concepts need to know to construct a CNN model are

1. Convolution Layer - The convolutional layer is the main building Block used in convolutional neural networks. It is a simple application of the filter that results in input activation. The innovation of the cumulative neural

network is the ability to automatically learn many parallel filters specific to a training data set in limitations of a particular predictive modeling problem, such as as an image classifier. The result is very specific features can be recognized anywhere in the input image.

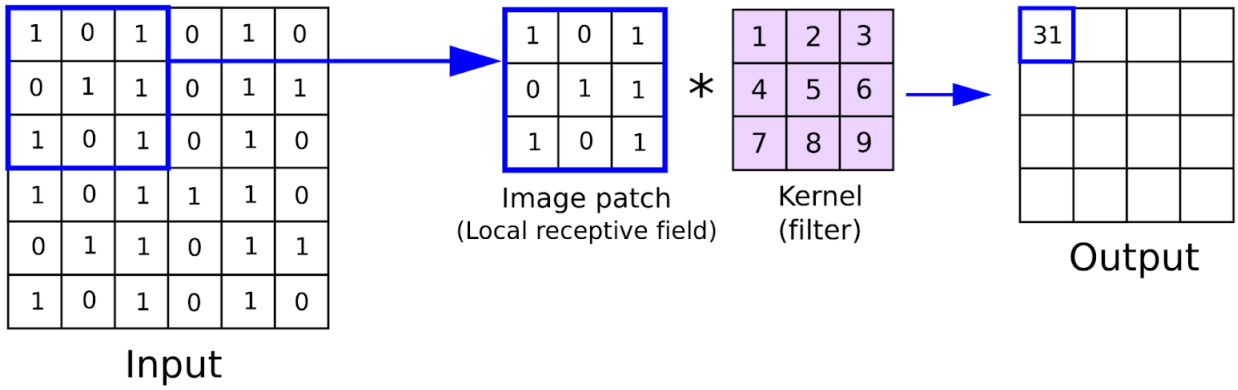

Figure 5.8: Convolution layer

2. ReLU Layer (Rectified Linear Unit) : ReLU stands for the Rectified Linear Unit for a non-linear operation. The output is f(x) = max(0,x).we use this because to introduce the non-linearity to CNN. ReLu(Rectified Linear Unit) is a linear function that will output the input itself, if input is positive,otherwise, it will output zero. As in the image 5.9 if input is less than 0 it return 0 and if input is greater that 0 it return the same as output.

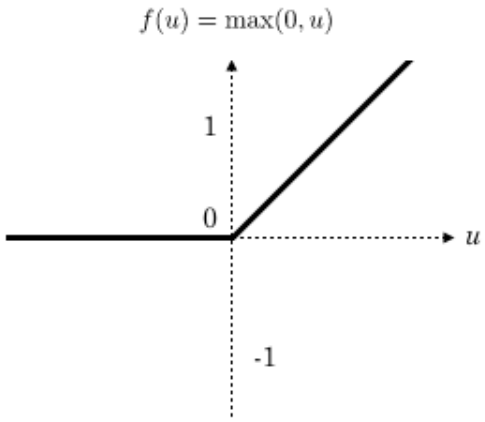

Figure 5.9: ReLu activation function

3.Pooling Layer (Max Pooling) : Pooling is done to shrink the image

matrix into a smaller size. Pooling is done after passing through a activation layer, here we used max pooling layer after convolution layer with ReLu activation function. In max pooling maximum value is considered in every patch of feature map. Hence input representation is down sampled.

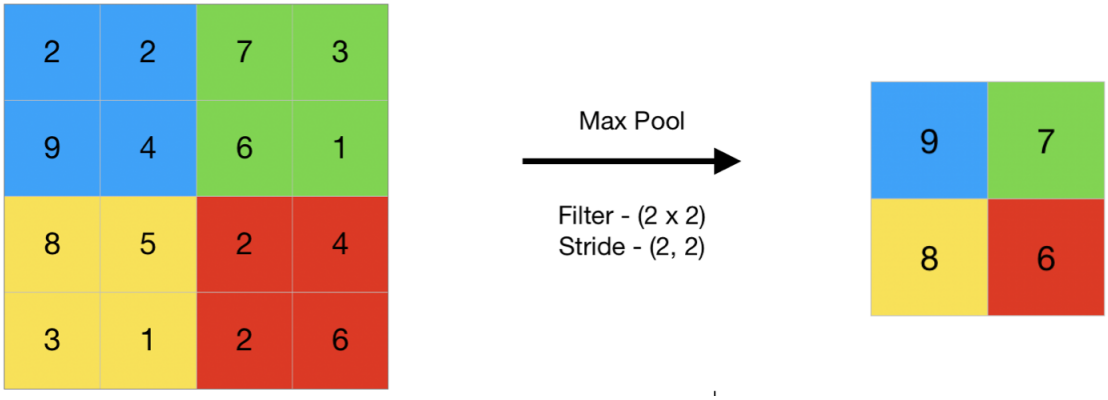

Figure 5.10: Maxpooling

4. Flattening : We flatten our entire matrix into a vector like a vertical one. So, that it will be passed to the input layer. We use flattening to convert all the resultant 2-D arrays obtained as output from pooled feature maps into a single linear vector. Here we use flattening layer before connecting to dense layer.The output obtained from flattening layer is given as input to dense layer.

5. Activation Layer: The activation layer controls how the signal streams starting with one layer then onto the next, imitating how neurons are terminated in our mind. output signals which are emphatically connected with past references would initiate more neurons, empowering signs to be spread all the more effectively for identification. CNN is perfect with a honest sort of complex enactment capacities to demonstrate signal propagation, the foremost basic capacity being the Rectified linear measure (ReLU), which is supported for its quicker preparing speed.

6. Fully Connected: The last layers inside the system are fully connected,implying that neurons of preceding layers are associated with every neuron in resulting layers. This emulates high level thinking where every single imaginable pathway from the input to output.

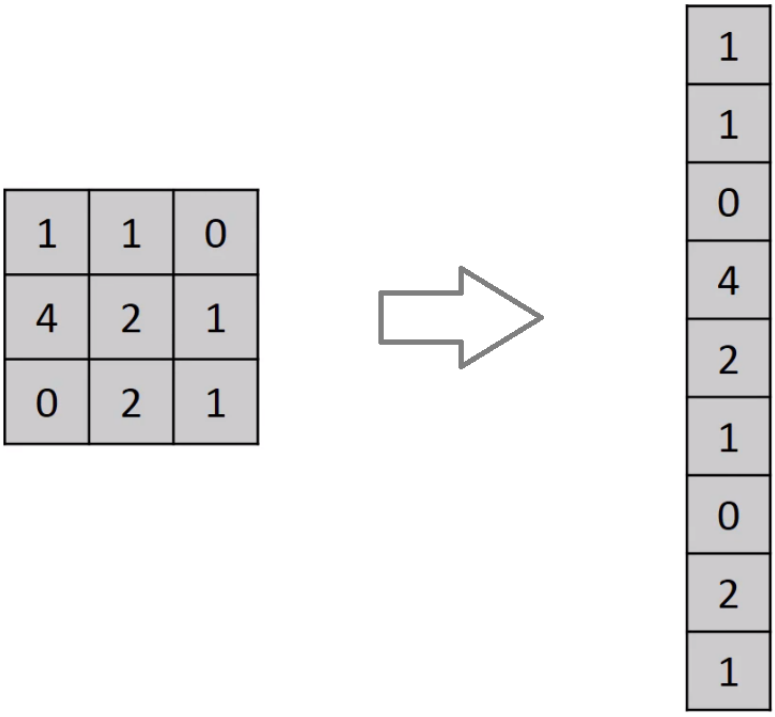

Figure 5.11: Flattening layer

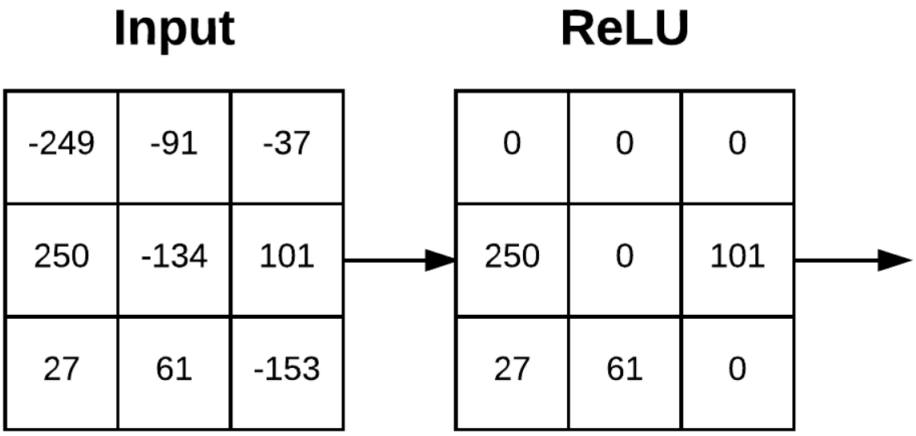

Figure 5.12: Activation layer

7. Loss (During Training) : When training the neural network, there is extra layer called the misfortune layer. This layer gives criticism to the neural network on whether it recognized sources of info effectively, and if not, the distance away its estimates were. This assists with controlling the neural system to reinforce the correct ideas since it trains. This is consistently the last layer during training. The loss function we used in the model is categorical cross entropy. While evaluating the epoches both accuracy and loss while be displayed in the logs.

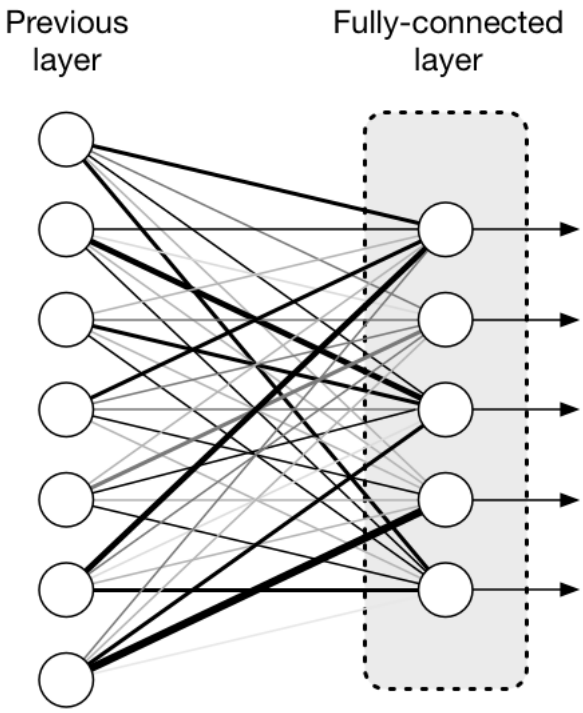

Figure 5.13: Fully connected layer

8. Adam Optimizer: Adam optimizer is all also called Adaptive moment estimator. It is one of the best optimizers. It combines momentum and RMSprop. Momentum helps in smoothing and RMSprop helps changing learning rate in a more efficient way. Here in this project we use adam optimizer.

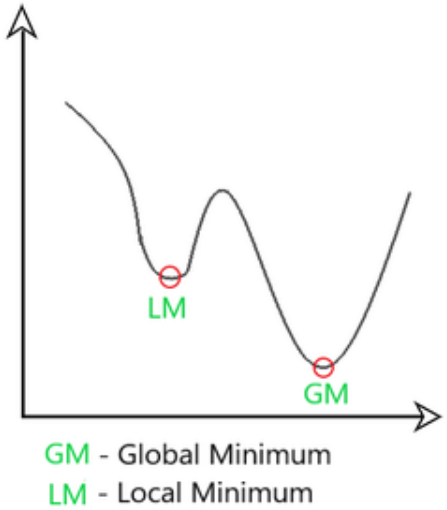

Figure 5.14: Adam Optimizer

5.3.2 Design of Test cases and scenarios

Testing is the process of executing a program with the aim of finding errors. To make our model perform well it should be error-free. If testing is done successfully it will remove all the errors from the software. The testing should be customised of the consumer. Program testing should be done by a third party since we will know how accurate our software is if it is done by a third party. All of the tests that will be carried out should be prepared before they are performed out. Testing should begin with simple parts and progress to larger components. Test Cases:

- In this project test cases will be giving input of different polynomial equations and the system should give the accurate value of given solved equation.
- To check whether the contours are generated or not.
- To check whether the given machine learning algorithm we used is applicable or not.

5.3.3 Forms

The proposed CNN project has been executed on Jupyter Notebook editor which can be used as an interface to implement Deep leaning models using the CNN algorithm. Two convolutional layers are used. We have used two max pooling layers. After one convolutional layer one max polling layer is used. Activation function is relu for all layers except for output dense layer. Relu activation function is used as it does not activate all the neurons at once. Neurons whose value is less than 0 are not considered as Relu outputs 0 value for values less than 0. It outputs the same value if value is greater than 0. In output layer that is dense layer softmax activation function is used. As it is multi classification problem softmax activation function is used.

HYPERPARAMETER	DESCRIPTION
No. of convolutional layers	2
No. of max pooling layers	2
Activation function	Relu
No. of epochs	10
Optimizer	Adam optimizer

Figure 5.15: Hyper parameters of CNN

5.3.4 Input Screens

The dataset we used is from kaggle. Kaggle is an online community of data scientists and machine learning practitioners. Kaggle allows users to find and publish data sets, explore and build models in a web-based data-science environment, work with other data scientists and machine learning engineers,and enter competitions to solve data science challenges. Input can be any equation.It can be a simple arithmetic equation or polynomial equation. Polynomial equation can be of any degree. It can be linear equation or quadratic equation or any equation of higher degree.

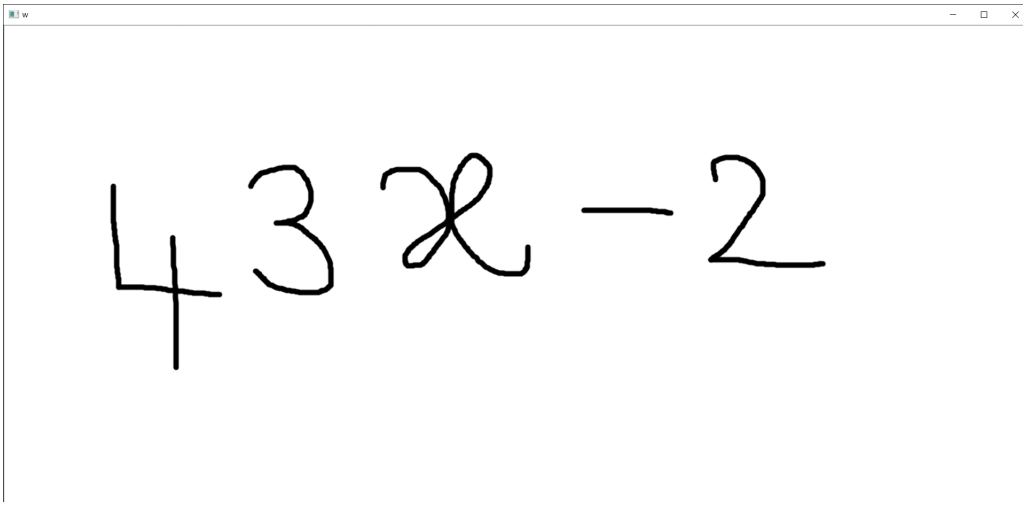

Figure 5.16: input linear equation

After taking the image of equation as input, contours are generated. Contours are simply a curve joining all the continuous points (along the boundary), having same color or intensity. The contours are a useful tool

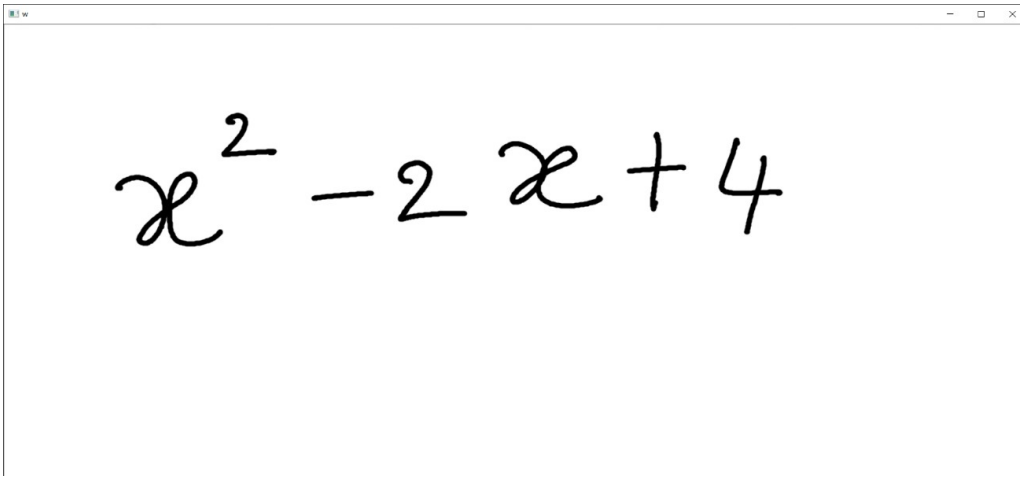

Figure 5.17: input quadratic equation

for shape analysis and object detection and recognition. For each digit and symbol contour is generated. Contours obtained for linear equation given as input(figure 5.16).

Figure 5.18: Contours obtained for linear equation

Contours obtained for quadratic equation given as input(figure 5.17)

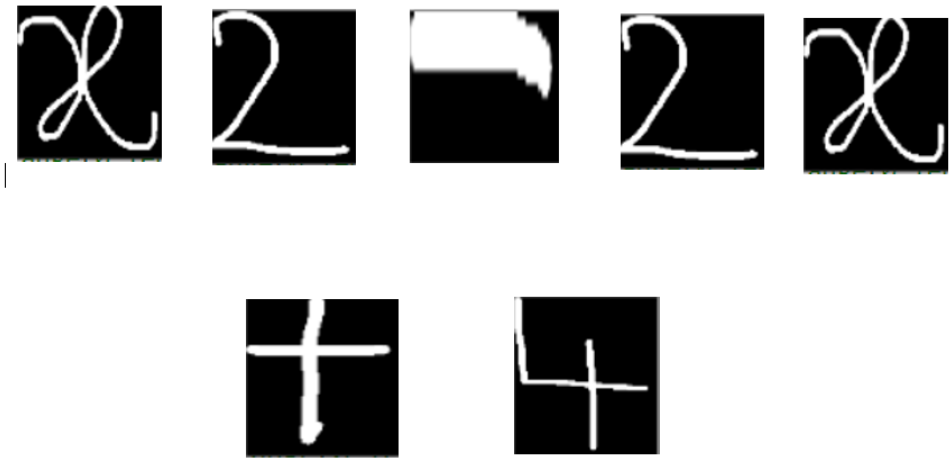

Figure 5.19: Contours obtained for quadratic equation

5.3.5 Result and Analysis

We ran the proposed model for 30 epochs and validation and training accuracy is presented in 5.20. The average accuracy of the proposed model is 98 percent approximately.

```
In [14]: model.fit(np.array(l),cat,shuffle=True,epochs=10)

Epoch 1/10
4895/4895 [==============================] - 66s 13ms/step - loss: 0.2403 - accuracy: 0.9355
Epoch 2/10
4895/4895 [==============================] - 58s 12ms/step - loss: 0.0752 - accuracy: 0.9791
Epoch 3/10
4895/4895 [==============================] - 62s 13ms/step - loss: 0.0545 - accuracy: 0.9847
Epoch 4/10
4895/4895 [==============================] - 63s 13ms/step - loss: 0.0438 - accuracy: 0.9878
Epoch 5/10
4895/4895 [==============================] - 63s 13ms/step - loss: 0.0388 - accuracy: 0.9892
Epoch 6/10
4895/4895 [==============================] - 60s 12ms/step - loss: 0.0351 - accuracy: 0.9905
Epoch 7/10
4895/4895 [==============================] - 61s 12ms/step - loss: 0.0322 - accuracy: 0.9912
Epoch 8/10
4895/4895 [==============================] - 62s 13ms/step - loss: 0.0291 - accuracy: 0.9921
Epoch 9/10
4895/4895 [==============================] - 61s 13ms/step - loss: 0.0283 - accuracy: 0.9925
Epoch 10/10
4895/4895 [==============================] - 64s 13ms/step - loss: 0.0268 - accuracy: 0.9930

Out[14]: <tensorflow.python.keras.callbacks.History at 0x1dc843ca3d0>
```

Figure 5.20: epocs runned and accuracy

5.3.6 Output Screens

After the contours are generated bounding rectangles are extracted. Bounding rectangles are location of contours in image. They highlight contours from the image.

```
rects [[159, 237, 168, 279], [366, 208, 128, 196], [563, 190, 226, 186], [604, 300, 58, 56], [674, 198, 50, 83], [865, 272, 13
8, 13], [1054, 193, 177, 170]]
bools [[0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 1, 1, 0, 0], [0, 0, 1, 0, 0, 0, 0], [0, 0, 1, 0, 0, 0, 0], [0,
0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0]]
dump_rects [[604, 300, 58, 56], [674, 198, 50, 83]]
final rects [[159, 237, 168, 279], [366, 208, 128, 196], [563, 190, 226, 186], [865, 272, 138, 13], [1054, 193, 177, 170]]
```

Figure 5.21: Bounding rectangles for contours of linear equation

Figure 5.21 shows the bounding rectangles that are obtained for the image given as a input where contours generated are figure 5.18.

```
rects [[166, 232, 163, 119], [206, 305, 30, 37], [252, 240, 31, 43], [326, 139, 83, 71], [462, 262, 94, 17], [591, 214, 105, 9
8], [742, 184, 158, 107], [767, 245, 34, 32], [815, 199, 39, 29], [936, 170, 89, 126], [1073, 179, 97, 153]]
bools [[0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0], [1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0], [0,
0, 0, 0, 0, 0, 1, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0]]
dump_rects [[206, 305, 30, 37], [252, 240, 31, 43], [767, 245, 34, 32], [815, 199, 39, 29]]
final rects [[166, 232, 163, 119], [326, 139, 83, 71], [462, 262, 94, 17], [591, 214, 105, 98], [742, 184, 158, 107], [936, 17
0, 89, 126], [1073, 179, 97, 153]]
```

Figure 5.22: Bounding rectangles for contours of quadratic equation

Bounding rectangles are the dimensions of the contours generated for each character or symbol in the equation. Figure 5.22 shows the bounding rectangles that are obtained for the image given as a input where contours generated are figure 5.19. Here we have considered the input that have eight characters in the equation. Hence there are eight final rectangles that are obtained.

```
In [16]: s=""
for i in range(len(train_data)):
    train_data[i]=np.array(train_data[i])
    train_data[i]=train_data[i].reshape(1,28,28,1)
    result=np.argmax(loaded_model.predict(train_data[i]), axis=-1)
    if(result[0]==10):
        s=s+'-'
    if(result[0]==11):
        s=s+'+'
    if(result[0]==12):
        s=s+'X'
    if(result[0]==13):
        s=s+'*'
    if(result[0]==0):
        s=s+'0'
    if(result[0]==1):
        s=s+'1'
    if(result[0]==2):
        s=s+'2'
    if(result[0]==3):
        s=s+'3'
    if(result[0]==4):
        s=s+'4'
    if(result[0]==5):
        s=s+'5'
    if(result[0]==6):
        s=s+'6'
    if(result[0]==7):
        s=s+'7'
    if(result[0]==8):
        s=s+'8'
    if(result[0]==9):
        s=s+'9'
print(s)

8X2-9X+5
```

Figure 5.23: Equation string that is generated for linear equation

After recognition, all the symbols or digits in the equation are appended in the form of a string. The output will initially be generated in the form of a string as shown in figure 5.23. Figure 5.23 shows the equation string formed when linear equation (figure 5.16) is given as input. Here initially '4' is recognized and is attached to empty string. Then '3' is recognized and added to string '4' which becomes '43'. Now 'x' is recognized and is added to string '43' which becomes '43x'. Then '-' and '2' are recognized and are added to string respectively where string becomes '43x-2'. Recognized string is given as output and is further evaluated for roots.

After recognition, all the symbols or digits in the equation are appended in the form of a string. The output will initially be generated in the form of a string as shown in figure 5.24. Figure 5.24 shows the equation string formed when quadratic equation (figure 5.17) is given as input. Here initially 'x' is recognized and is attached to empty string. Then '2' is recognized and

```
In [179]: s=""
for i in range(len(train_data)):
    train_data[i]=np.array(train_data[i])
    train_data[i]=train_data[i].reshape(1,28,28,1)
    result=np.argmax(loaded_model.predict(train_data[i]), axis=-1)
    if(result[0]==10):
        s=s+'-'
    if(result[0]==11):
        s=s+'+'
    if(result[0]==12):
        s=s+'X'
    if(result[0]==13):
        s=s+'*'
    if(result[0]==0):
        s=s+'0'
    if(result[0]==1):
        s=s+'1'
    if(result[0]==2):
        s=s+'2'
    if(result[0]==3):
        s=s+'3'
    if(result[0]==4):
        s=s+'4'
    if(result[0]==5):
        s=s+'5'
    if(result[0]==6):
        s=s+'6'
    if(result[0]==7):
        s=s+'7'
    if(result[0]==8):
        s=s+'8'
    if(result[0]==9):
        s=s+'9'
print(s)

X2-2X+4
```

Figure 5.24: Equation string that is generated for quadratic equation

added to string 'x' which becomes 'x2'. Now '-' is recognized and is added to string 'x2' which becomes 'x2-'. Then '2', 'x', '+' and '4' are recognized and are added to string respectively where string becomes 'x2-2x+4'. Recognized string is given as output and is further evaluated for roots.

After the string is generated it is again converted to list, then digits are appended and are converted to numbers accordingly. Then highest degree is identified. A zero-list of size of highest degree is generated. Co-efficient according to power are updated to list. Obtained list is used to generate roots.

```
In [14]: res=[]
res=numberlist(list(s))
print(res)
solution(res)

['43', 'X', '-', '2']
43 X - 2
degree is  1
['4', '3', 'X', '-', '2']

Out[14]: array([0.04651163])
```

Figure 5.25: Final output that is generated for linear equation

The string that is generated is evaluated and is given as final output as shown in the figure 5.25 for linear equation.

```
In [183]: solution(list(s))

          X^2 - 2 X + 4
          degree is  2
          ['X', '2', '-', '2', 'X', '+', '4']
          [ 1. -2.  4.]
          [1.+1.73205081j 1.-1.73205081j]

Out[183]: array([1.+1.73205081j, 1.-1.73205081j])
```

Figure 5.26: Final output that is generated for quadratic equation

The string that is generated is evaluated and is given as final output as shown in the figure 5.26 for quadratic equation. Roots of the polynomial equation is generated as output.

CHAPTER 6

Conclusions

6.1 Conclusion

To simplify the math, the main task done is the feature extraction from the image and recognition with the help of the CNN model. If the CNN model classifies correctly all of the segmented images then this will generate the correct polynomial equation. In the proposed CNN based architecture there are two convolution and max pooling layers. This is a successful representation of the state of the art. Experimentally, it is observed the testing accuracy of the model is about 98 percent. In comparison with other deep-learning approaches, the implemented deep-learning model has better predictive ability in terms of both accuracy and loss. The required time to train the model was much less than that of other machine-learning approaches. Using this model the equation is recognised successfully with a good rate of accuracy. Also the data set we used contains images of every class, where it contains different images in each class. That is each image have different hand writing.Hence this project works for almost every hand writing. But there are also limitations to this project. One of the limitation is that if we write two characters or digits or symbols with no space that is if two symbols are collided with each other, then they are identified as a single symbol. The background of the image must be clean. That is only the characters, symbols or digits should be present. If there are unnecessary lines present in background they would also be recognized as some class that is matched closely. One of the main advantage of this project is that any person can easily use this as the process is easy and does not require any advance knowledge of the technology.

6.2 Future scope

Here we demonstrate a model which can recognize handwritten polynomial equation and solve that equation. Later it can be extended for single quadratics and also series of quadratics. As seen from the results of the experiment, CNN proves to be far better than other classifiers. The results can be made more accurate with more convolution layers and more number of hidden neurons. It can completely abolish the need for typing. We can also work on the limitations. In future we will try to improve the accuracy and try to make the system workable for multiple mathematical formula simultaneously. We will also work to build a user friendly interface, where user can directly take images or upload images then roots are directly obtained as output. All of the above mentioned points can be used for future research.

REFERENCES

[1] Azzeddine Lazrek Widad Jakjoud. "Segmentation method of offline mathematical symbols". In: *International Conference on Multimedia Computing and Systems (ICMCS)* (2011).

[2] Jyothy R.L et al. Shilpa C Vijayan. "Histogram Based Connected Component Analysis for Character Segmentation". In: *International Journal of Scientific and Research Publications* 6 (2016).

[3] *Course note of CNN Stanford.* URL: `http://cs231n.github.io/convolutional-networks/`.

[4] Garg V.K. Cipolla R. Shankar S. "Deep-carving: Discovering visual attributes by carving deep neural nets". In: *IEEE Conference on Computer Vision and Pattern Recognition* (2015), pp. 3403–3412.

[5] Sonu Agrawal Pooja Kamavisdar Sonam Saluja. "A Survey on Image Classification Approaches and Techniques". In: *international Journal of Advanced Research in Computer and Communication Engineering* 2 (2013).

[6] Eric Lecolinet Richard G.Casey. "A survey of methods and strategies in character segmentation". In: (1996).