

FOURTH YEAR SEMINAR REPORT

A Technical Review on handwritten Expression Solver from Images

Submitted in partial fulfilment of the degree of Bachelor of Technology
Rajasthan Technical University



By

Harsh Anand
(PIET16CE033)

DEPARTMENT OF COMPUTER ENGINEERING
POORNIMA INSTITUTE OF ENGINEERING & TECHNOLOGY, JAIPUR
(Academic Year 2019-20)

FOURTH YEAR SEMINAR REPORT

A Technical Review on handwritten Expression Solver from Images

Submitted in partial fulfilment of the degree of Bachelor of Technology
Rajasthan Technical University



By

Harsh Anand
(PIET16CE033)

DEPARTMENT OF COMPUTER ENGINEERING
POORNIMA INSTITUTE OF ENGINEERING & TECHNOLOGY, JAIPUR
(Academic Year 2019-20)



RAJASTHAN TECHNICAL UNIVERSITY
POORNIMA INSTITUTE OF ENGINEERING & TECHNOLOGY, JAIPUR

CERTIFICATE

This is to certify that Final Year Practical Training Seminar Report entitled "**A Technical Review on Handwritten Expression Solver from Images**" has been submitted by "Harsh Anand (CE/16/033)" for partial fulfilment of the Degree of Bachelor of Technology of Rajasthan Technical University. It is found satisfactory and approved for submission.

Date: 24.3.20

Mr. Deepak Mood
Head,
Dept. of Comp Egg
PIET, Jaipur

Rd. Dinesh Goyal
Director,
PIET, Jaipur

DECLARATION

I hereby declare that the seminar report entitled "A TECHINAL REVIEW ON HANDWRITTEN EXPRESSION SOLVER FROM IMAGES" was carried out and written by me under the guidance of Ms. Shruti Bijawat, Assistant Professor, Department of Computer Engineering, Poornima Institute of Engineering & Technology, Jaipur. This work has not been previously formed the basis for the award of any degree or diploma or certificate nor has been submitted elsewhere for the award of any degree or diploma.

Place: Jaipur

Harsh Anand

Date: 24-03-20

PIET16CE033

ACKNOWLEDGEMENT

A Technical Review on Handwritten Expression Solver from Images is such a vast coverage cannot be realized without help from numerous sources and people in the organization. I am thankful to **Dr. Dinesh Goyal, Director, for** providing me a platform to carry out such a training successfully.

I am also very grateful to **Mr. Deepak Moud (HOD, CE)** for his kind support.

I would like to take this opportunity to show my gratitude towards **Dr. Rekha Jain & Dr. Megha Gupta (Seminar Coordinator, PTS)** who helped me in successful completion of my Fourth Year Seminar Topic. They have guided, motivated & were source of inspiration for me to carry out the necessary proceedings for the training to be completed successfully.

I am thankful to **Seminar Guide Ms. Shruti Bijawat** for her kind support and providing me expertise of the domain to develop the project.

I am also privileged to have **Mr. Deepak Moud, Dr. Puneet Mathur** who has/have flourished me with his/her/their valuable facilities without which this work cannot be completed.

I would also like to express my hearts felt appreciation to all of my friends whose direct or indirect suggestions help me to develop this project [and to entire team members for their valuable suggestions.

Lastly, thanks to all faculty members of Computer Engineering department for their moral support and guidance.

Submitted by:

Harsh Anand

ABSTRACT

Text or Character recognition from images is a very important active research area which helps to develop a computer application that has the ability to automatically fetch the text or character from images (Text or Character in images be either Handwritten or Computer Generated). Now a day's there is huge increasing in demand of storing the information that are available on paper documents in to a computer readable form for further or later use. A very simple way to store the information from these paper documents in to computer system is with the help of scanning the documents and then store them as in the form of images. However, when we want to reuse this stored information, it is going to be very difficult to read the individual contents from the images and searching the contents form these documents line by-line or word-by-word. There are certain challenges involved in this: font characteristics of the characters, quality of the images and detecting the handwritten character (either operator or operand) with accurate result. Due to these type of challenges, computer is unable to recognize the handwritten characters while reading them. So, there is a need of handwritten character recognition mechanisms just to perform document image analysis which transforms documents by segmenting each handwritten character from image then predict the segmented character and store all the segmented character in single variable in electronic format. In this paper, we have reviewed and analysed different methods for text recognition and Handwritten expression recognition from images. The objective of this review paper is to summarize the well-known methods that we used for recognition purpose that helps for better understanding of the reader.

Keyword:

Handwritten Expression Solver, Electronic format, Handwritten text recognition, font characteristics, model for detecting handwritten characters.

Implementation Software and Hardware:

Python, Spyder, Anaconda, Laptop, Various Libraries like scipy, smipy, tensorflow, pytesseract, keras, etc., MS-Paint.

TABLE OF CONTENTS

CHAPTER NO.	TOPICS	PAGE NO.
	TITLE PAGE	
	CERTIFICATE	1
	CANDIDATE'S DECLARATION	2
	ACKNOWLEDGEMENT	3
	ABSTRACT	4
	LIST OF CONTENTS	5-6
	LIST OF FIGURES	7-8
1	INTRODUCTION	9-11
	1.1 Research Aim 1.2 Problem Description 1.3 Significance of Topic	
2	TECHNOLOGY SPECIFICATION	12-16
	2.1 Language Learned 2.2 Tools or Techniques	
3	TOPIC DESCRIPTION	17-20
	3.1 Problem Statement 3.2 Software & Hardware Requirements	
4	PROJECT IMPLEMENTATION	21-28
	4.1 Sprint Backlog-1 4.2 Sprint Backlog-2 4.3 Sprint Backlog-3 4.4 Sprint Backlog-4	

5	RESULT & FUTURE SCOPE	29-32
	5.1 Future Scope 5.2 Result 5.3 Working Architecture	
6	LIMITATIONS	33
	REFERENCES	34

LIST OF FIGURES

S. NO.	FIGURE TITLE	PAGE No.
1.	Fig 4.01 Expressions (Store in png format)	21
2.	Fig 4.02 Sample code for fetch and solve the expression from Image	22
3.	Fig 4.03 Result after solving the expression from Image	22
4	Fig 4.04 Expressions (Store in png format)	23
5.	Fig 4.05 UI for our Project	23
6.	Fig 4.06 Sample Code for UI connect with flask	24
7.	Fig 4.07 Sample Code of UI Integrate with flask	25
8.	Fig 4.08 Sample Code for UI Integrate with flask	25
9.	Fig 4.09 Expression (Store in png format)	26
10.	Fig 4.10 UI for our project	26
11.	Fig 4.11 Handwritten Expression (Store in png format)	27
12.	Fig 4.12 Sample code for training model for handwritten expression	27
13.	Fig 4.13 Sample Flask app (handwritten)	28
14.	Fig 4.14 UI for Handwritten Expression Solver	28
15.	Fig 5.01 Result 1	30
16.	Fig 5.02 Result 2	30
17.	Fig 5.03 Result 3	31

18.	Fig 5.04 Result 4	32
19.	Fig 5.05 Working Architecture of Handwritten Expression Solver	32

Chapter 1: Introduction

1.1 Research Aim

Rapid growth of data comes with a challenge of sorting and analysing them, where raw data exists in graphical form, textual form or in images. Data science and machine learning finds its application in various fields like stock market, recommendation systems, image processing, aerial photography, military, weather forecasting etc. As we all know, there is a lot of data present in the form of paper pages that can be of any books, thesis and various historical and mythological books. And due to change in atmosphere or due to the mishandling of the pages it is very difficult to store that valuable data and therefore it is a requirement of any software that can convert that data into the digital data. And after converting the data it would be very useful that can be shared with anyone without any distortion of the data. That means we get the form the documents line-by-line and word-by-word completely. The difficulty arises due to the difference in the font characteristics of the characters in the paper document and the font characteristics of the character in the computer system. There are many existing Optical Character Recognition solutions are commonly used. However, there is a still challenging problem for recognition of handwritten text. This paper proposes an efficient approach to detect the handwritten text using the convolution neural network (CNN) and OpenCV. We have used the dataset publically available NIST datasets which contains thousands of handwritten characters.

As we all know, there is a lot of data present in the form of paper pages that can be of any books, thesis and various historical and mythological books. And due to change in atmosphere or due to the mishandling of the pages it is very difficult to store that valuable data and therefore it is a requirement of any software that can convert that data into the digital data. And after converting the data it would be very useful that can be shared with anyone without any distortion of the data. That means we get the form the documents line-byline and word-by-word completely. The difficulty arises due to the difference in the font characteristics of the characters in the paper document and the font characteristics of the character in the computer system. There are many existing Optical Character Recognition solutions are commonly used.

1.2 Problem Description

Identifying handwritten expression is the most difficult thing to do specially in the case of some digits like ‘/’ is very similar to ‘1’ in the math symbol and same case with ‘8’ and ‘0’ and so on. Identifying the alphabets in the math expression is also very difficult task to do like it is very difficult to identify the difference between ‘0’ and ‘o’ at the time to check for the constant variables in order to fetch the expression and solve it using the appropriate method of math and same with the case of two and more than three expressions. And also calculating the roots of quadratic expression specially in the case of negative roots is also very difficult thing to do. And also working with integration and differentiation in the case of handwritten is also a challenge to solve. Not only the in the case of simple problems but also in the case of some complex problems of same. There is no such tool available now which can work on all types of handwritten problems of math. And expression can also be in the form of images like there is an image in which an expression is specified so we have to extract the expression from the image and identify the operators and the digits and provide the correct solution of the expression. So there is no such application available on the internet which can work on both the cases either for handwritten or in image form.

1.3 Significance of the topic

For centuries, handwritings had been the most common way of education. However, using handwritings in this education world is still something very appreciable and which remains possible, especially with the emergence of devices that rely on digital pens as an input method. Developing those devices was accompanied with the development of recognition systems capable of converting texts from our natural handwriting and vice-versa, into languages understandable by computers. This paper is based on Project of solving the mathematical expression which can be handwritten or in image. First this will extract the expression from the image then this will identify the digits and operators. After identifying the digit and operators through the open cv and deep learning model. Now the expression is solved using python. Text or Character recognition from images is a very important active research area which helps to develop a computer application that has the ability to automatically fetch the text or character from images (Text or Character in images be either Handwritten or Computer Generated). Now a day’s there is huge

increasing in demand of storing the information that are available on paper documents in to a computer readable form for further or later use. A very simple way to store the information from these paper documents in to computer system is with the help of scanning the documents and then store them as in the form of images. However, when we want to reuse this stored information, it is going to be very difficult to read the individual contents from the images and searching the contents from these documents line by-line or word-by-word. There are certain challenges involved in this: font characteristics of the characters, quality of the images and detecting the handwritten character (either operator or operand) with accurate result. Due to these type of challenges, computer is unable to recognize the handwritten characters while reading them. So, there is a need of handwritten character recognition mechanisms just to perform document image analysis which transforms documents by segmenting each handwritten character from image then predict the segmented character and store all the segmented character in single variable in electronic format. In this paper, we have reviewed and analysed different methods for text recognition and Handwritten expression recognition from images. The objective of this review paper is to summarize the well-known methods that we used for recognition purpose that helps for better understanding of the reader.

Chapter 2: Technology Specification

2.1 Language Learned

Python - It is an interpreted language, high-level and general-purpose programming language, it had been first released in 1991. Python language is made by Guido van Rossum and Python's design philosophy emphasizes code readability with its notable use of serious whitespace or indentation. It is language constructs and follows object-oriented approach that help the programmers to aim for write clear logical code for small and large-scale projects. Python supports multiple programming paradigms, including procedural, object-oriented, and functional programming.

There are two major version of Python:

- Python 2
- Python 3

Python was designed for readability, and has some similarities to English language with influence from mathematics. Python uses new lines to finish a command, as against other programming languages which frequently use semicolons or parentheses. Python relies on indentation, using whitespace, to define scope; like the scope of loops, functions and classes. Other programming languages often use curly-brackets for this purpose.

Python is used in our project because it is simple to learn, it provides better functionalities to work with various new technologies like Open-CV, Machine Learning, Deep Learning, Artificial Intelligence, Flask etc.

It is used for Web development, Software development, mathematics, Scripting. Python often wont to develop different applications like web applications, graphic interface based applications, software development application, scientific and numeric applications, schedule, Games and 3D applications and other business applications.

2.2 Tools or Techniques

Flask – It is used for providing the connection between the interface of the UI and the python code. Python code is used for detecting the handwritten text or solving the detected handwritten expression. Flask may be a lightweight WSGI web application framework. it's designed to form

getting started quick and straightforward, with the power to proportion to complex applications. It began as an easy wrapper around Werkzeug and Jinja and has become one among the foremost popular Python web application frameworks.

Flask offers suggestions, but doesn't enforce any dependencies or project layout. it's up to the developer to settle on the tools and libraries they need to use. There are many extensions provided by the community that make adding new functionality easy.

OpenCV – OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to supply a standard infrastructure for computer vision applications and to accelerate the utilization of machine perception within the commercial products. Being a BSD-licensed product, OpenCV makes it easy for businesses to utilize and modify the code.

The library has quite 2500 optimized algorithms, which incorporates a comprehensive set of both classic and state-of-the-art computer vision and machine learning algorithms. These algorithms are often wont to detect and recognize faces, identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch images together to supply a high resolution image of a whole scene, find similar images from a picture database, remove red eyes from images taken using flash, follow eye movements, recognize scenery and establish markers to overlay it with augmented reality, etc. OpenCV has quite 47 thousand people of user community and estimated number of downloads exceeding 18 million. The library is employed extensively in companies, research groups and by governmental bodies.

Along with well-established companies like Google, Yahoo, Microsoft, Intel, IBM, Sony, Honda, Toyota that employ the library, there are many startups like Applied Minds, VideoSurf, and Zeitera, that make extensive use of OpenCV. OpenCV's deployed uses span the range from stitching streetview images together, detecting intrusions in surveillance video in Israel, monitoring mine equipment in China, helping robots navigate and devour objects at Willow Garage, detection of swimming bath drowning accidents in Europe, running interactive art in Spain and ny , checking runways for debris in Turkey, inspecting labels on products in factories round the world on to rapid face detection in Japan.

It has C++, Python, Java and MATLAB interfaces and supports Windows, Linux, Android and Mac OS. OpenCV leans mostly towards real-time vision applications and takes advantage of

MMX and SSE instructions when available. A full-featured CUDA and OpenCL interfaces are being actively developed immediately. There are over 500 algorithms and about 10 times as many functions that compose or support those algorithms. OpenCV is written natively in C++ and features a templated interface that works seamlessly with STL containers.

Pytesseract – Python-tesseract is an optical character recognition (OCR) tool for python. That is, it'll recognize and “read” the text embedded in images.

Python-tesseract may be a wrapper for Google's Tesseract-OCR Engine. it's also useful as a stand-alone invocation script to tesseract, because it can read all image types supported by the Pillow and Leptonica imaging libraries, including jpeg, png, gif, bmp, tiff, et al. Additionally, if used as a script, Python-tesseract will print the recognized text rather than writing it to a file.

Tesseract was within the top three OCR engines in terms of character accuracy in 1995. it's available for Linux, Windows and Mac OS X. However, thanks to limited resources it's only rigorously tested by developers under Windows and Ubuntu.

Tesseract up to and including version 2 could only accept TIFF images of straightforward one-column text as inputs. These early versions didn't include layout analysis, then inputting multi-column text, images, or equations produced garbled output. Since version 3.00 Tesseract has supported output text formatting, hOCR positional information and page-layout analysis. Support for variety of latest image formats was added using the Leptonica library. Tesseract can detect whether text is monospaced or proportionally spaced.

Tesseract can process right-to-left text like Arabic or Hebrew, many Indic scripts also as CJK quite well. Accuracy rates are shown during this presentation for Tesseract tutorial at DAS 2016, Santorini by Ray Smith.

Tesseract is suitable to be used as a backend and may be used for more complicated OCR tasks including layout analysis by employing a frontend like OCRopus.

Tesseract's output will have very poor quality if the input images aren't preprocessed to suit it: Images (especially screenshots) must be scaled up such the text x-height is a minimum of 20 pixels, any rotation or skew must be corrected or no text are going to be recognized, low-frequency changes in brightness must be high-pass filtered, or Tesseract's binarization stage will destroy much of the page, and dark borders must be manually removed, or they're going to be

misinterpreted as characters.

CNN - A Convolutional Neural Network (ConvNet/CNN) may be a Deep Learning algorithm which may absorb an input image, assign importance (learnable weights and biases) to varied aspects/objects within the image and be ready to differentiate one from the opposite. The pre-processing required during a ConvNet is far lower as compared to other classification algorithms. While in primitive methods filters are hand-engineered, with enough training, ConvNets have the power to find out these filters/characteristics.

The architecture of a ConvNet is analogous thereto of the connectivity pattern of Neurons within the Human Brain and was inspired by the organization of the visual area. Individual neurons answer stimuli only during a restricted region of the field of vision referred to as the Receptive Field. a set of such fields overlap to hide the whole visual cortex.

Keras - Keras is an open-source neural-network library written in Python. it's capable of running on top of TensorFlow, Microsoft Cognitive Toolkit, R, Theano, or PlaidML. Designed to enable fast experimentation with deep neural networks, it focuses on being user-friendly, modular, and extensible. it had been developed as a part of the attempt of project ONEIROS (Open-ended Neuro-Electronic Intelligent Robot Operating System), and its primary author and maintainer is François Chollet, a Google engineer. Chollet is also the author of the Xception deep neural network model. In 2017, Google's TensorFlow team decided to support Keras in TensorFlow's core library. Chollet explained that Keras was conceived to be an interface instead of a standalone machine learning framework. It offers a higher-level, more intuitive set of abstractions that make it easy to develop deep learning models no matter the computational backend used. Microsoft added a CNTK backend to Keras also, available as of CNTK v2.0.

Keras contains numerous implementations of commonly used neural-network building blocks like layers, objectives, activation functions, optimizers, and a number of tools to form working with image and text data easier to simplify the coding necessary for writing deep neural network code. The code is hosted on GitHub, and community support forums include the GitHub issues page, and a Slack channel. In addition to plain neural networks, Keras has support for

convolutional and recurrent neural networks. It supports other common utility layers like dropout, batch normalization, and pooling. Keras allows users to productize deep models on smartphones (iOS and Android), on the online , or on the Java Virtual Machine. It also allows use of distributed training of deep-learning models on clusters of Graphics processing units (GPU) and tensor processing units (TPU) principally in conjunction with CUDA.

Chapter 3: Topic Description

For centuries, handwritings had been the most common way of education. Nowadays, computers and the Internet are the indispensable way of modern communications; turning the planet into a little town. However, using handwritings in this education world is still something very appreciable and which remains possible, especially with the emergence of devices that rely on digital pens as an input method. Developing those devices was accompanied with the development of recognition systems capable of converting texts from our natural handwriting and vice-versa, into languages understandable by computers. This Project is solving the mathematical expression which can be handwritten or in image. First this will extract the expression from the image then this will identify the digits and operators. After identifying the digit and operators through the open cv and deep learning model. Now the expression is solved using python. This project can also solve the expressions containing the constant variables so this project is calculating the value of constant variables using appropriate method of mathematics. This will also provide the solution of the integration and differentiation problem with and without limits. Not only the single expressions but also the multiple expression can also be solved by this application. If there is an expression of degree two, then this will have solved it and calculate the multiple solution up to any possible limit. Not only the expression but also the power can also be calculated up to any limit and it can also solve the Boolean expression like $>$, $<$, $==$ operators. This Project will identify the handwritten Mathematical expression identify the operators and operands from the handwritten expression and provide the solution. The handwritten expression can be in the form of images, written in any format like .txt, docs.

Many tools are available to achieve this task. However, most of those tools require some predefined knowledge to use them. Latex and MathML, for example, require knowledge of predefined sets of key words to declare the variables to use them in the math's expression. Other tools, such as Math Type, depend on a visual environment to add symbols using the input through the keyboard which is a time consuming process. Our research focuses on the recognition of human handwritten mathematical expressions (ME).

3.1 Problem Statement:

Identifying handwritten expression is the most difficult thing to do specially in the case of some digits like ‘/’ is very similar to ‘1’ in the math symbol and same case with ‘8’ and ‘0’ and so on. Identifying the alphabets in the math expression is also very difficult task to do like it is very difficult to identify the difference between ‘0’ and ‘o’ at the time to check for the constant variables in order to fetch the expression and solve it using the appropriate method of math and same with the case of two and more than three expressions. And also calculating the roots of quadratic expression specially in the case of negative roots is also very difficult thing to do. And also working with integration and differentiation in the case of handwritten is also a challenge to solve. Not only the in the case of simple problems but also in the case of some complex problems of same. There is no such tool available now which can work on all types of handwritten problems of math. And expression can also be in the form of images like there is an image in which an expression is specified so we have to extract the expression from the image and identify the operators and the digits and provide the correct solution of the expression. So there is no such application available on the net which can work on both the cases either for handwritten or in image form. There is some application available but accuracy is the major issues with them because if we not able to identify the digits then the results of calculation will always be wrong no matter any method we apply so major challenge is to identify the most accurate method by applying the all available machine learning and deep learning algorithms and after that select one which give us the higher accuracy then all. And also finding out the dataset is also the challenge for us because there is no large dataset available on internet. And because dataset is collected from multiple resources so that it may be differ. Now a day there is no such tools are available by which students can use it for getting the answer of any handwritten expression or any expression or writing any expression through keyboard input and get the answer. Like there is an assignment given to any student of math’s and he/she have to check the answers of math problems without checking the solution of the question it should show directly the answer so that students can verify their answers. There are calculators available for normal calculation even for scientific calculators are also available for the some hard problems through which we have to provide the inputs through the buttons and then only it provides us the result but the multiple problems with the calculator and there are as follows:- first problem with the calculator is that it will only work for simple arithmetic problems and some advance calculator can work with the trigonometric

problems but it won't work with the linear expression and quadratic expression for calculating the values for constant variables. Second problem with the calculator is that it will only work for input data it will not work for the handwritten data. Third problem with the calculator is that it will take input directly it won't take an input image on which an expression is written and calculator is not capable enough to fetch the expression and provide the solution. Fourth problem with the calculator is that it is costly specially the scientific calculators are costlier than others. And there is such application available in the form of a website or app which is easily available on the internet and can solve all three types of problems like handwritten and image format and input driven on one website. Like if student have any doubt about any question then he simply have to write in a paper on his own language and click the photo of this handwritten expression and now upload this photo to the application named as handwritten expression solver and not this tool which extract the expression from the image and display it on the screen so that user can identify if it is the same expression he has written and after clicking on the submit button this will show the results on the same screen same goes with the rest of two applications. Like we are using the open cv in this application so there are many problems with the image like images may contain some noise which can cause problem in extraction like if an image contains the dot then this tool will extract it send it to the model for prediction and then our model will predict it as either zero or eight and same with the problem if there is any scratch on the image then our model will predict it as one or something so this will fall our calculation wrong and will decrease the accuracy of our application. When designing the user interface of the application there is one more problem that is the library because some large library may cause the time complexity to increase. Being able to input a mathematical expression into a digital document could be a difficult process. Tools, like LATEX or Math ML), aim at making it a neater and more convivial process. However, complex expressions require much efforts and time. employing a digital pen presents a more natural thanks to input such expressions into digital documents. Some researchers are emerging in this area, especially on subclasses of mathematical expressions with some promising results. Most of research works consider recognition of mathematical expressions as a set of subtasks to perform different steps of the recognition process like digit recognition etc. Though, a main drawback comes from the fact that any error at any step will be automatically inherited to the next step, requiring further processing to be sure of a good and correct recognition results. Our contribution to the domain of handwritten mathematical expressions recognition is

to perform a simultaneous segmentation, recognition and interpretation and solution of mathematical expressions. Specifically, the classifier wont to recognize the essential symbols is predicated on a worldwide learning method allowing the system to find out symbols directly from expressions rather than employing a pre-trained classifier. There are solution already available on the internet but there may be the possibility that there can be multiple possible answers of the same question then in this case student get confuse due to answers unmatched. But this application will provide the all the possible answers of the same question so that student will not get confused by the multiple answers and this application will also provide the answers up to any limit like in imaginary form as well.

3.2 Software and Hardware Requirements

- Python
- Spyder
- Anaconda
- Laptop
- Various Libraries like scipy, smipy, tensorflow, pytesseract, keras, etc.
- Sublime Text
- Chrome Browser
- MS-Paint
- Google Chrome Browser

Chapter 4: Project Implementation

4.1 Sprint Backlog- 1

Our Sprint Backlog-1 is basically based on the understand the requirements of tools & technologies, understand the future scope of the project, select the appropriate language to work on the project, install the required libraries to configure the system and create a model that extract simple expression from the image and solve that expression extracted from image. For starting phase generate the expression in image with the help of MS-Paint software. After getting the result test the accuracy of the program.

Expressions (Store in png format)

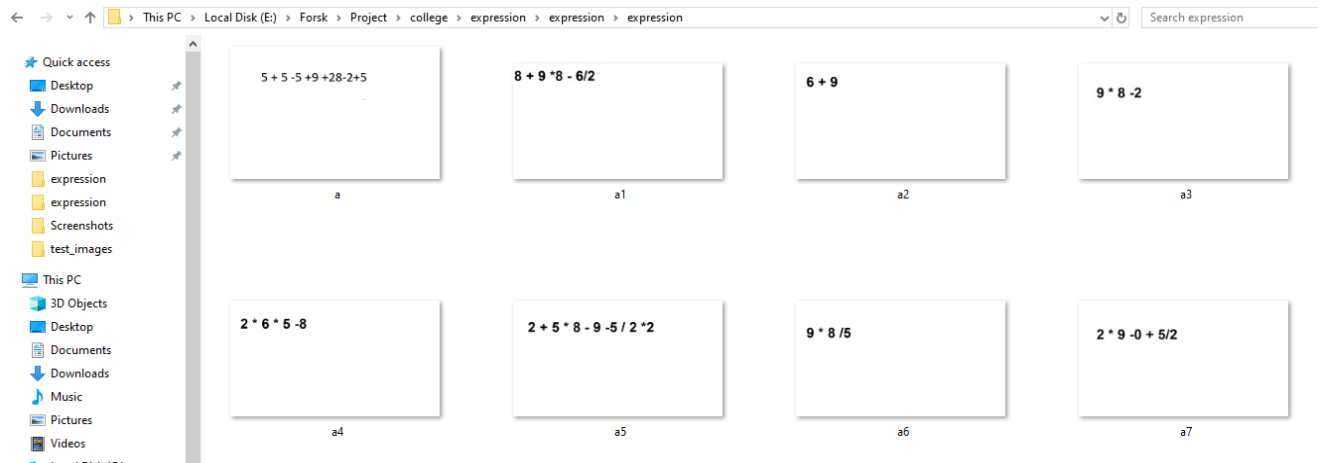
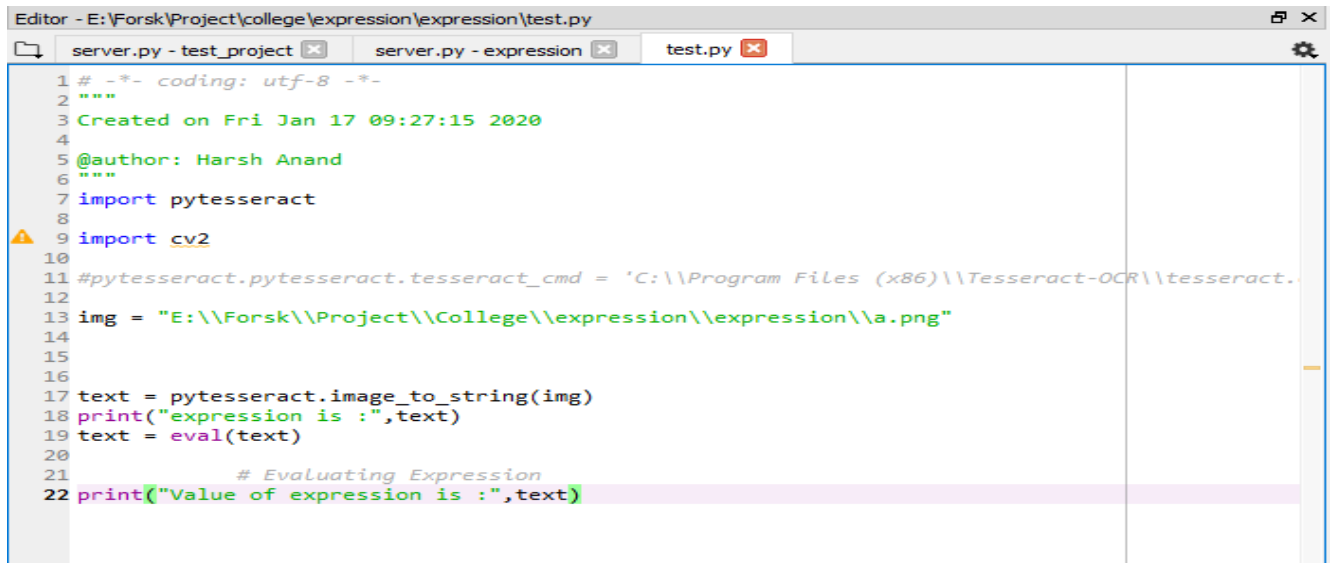


Fig 4.01

Sample code for fetch and solve the expression from Image



```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Fri Jan 17 09:27:15 2020
4
5 @author: Harsh Anand
6 """
7 import pytesseract
8
9 import cv2
10
11 #pytesseract.pytesseract.tesseract_cmd = 'C:\\Program Files (x86)\\Tesseract-OCR\\tesseract.
12
13 img = "E:\\Forsk\\Project\\College\\expression\\expression\\a.png"
14
15
16
17 text = pytesseract.image_to_string(img)
18 print("expression is :",text)
19 text = eval(text)
20
21 # Evaluating Expression
22 print("Value of expression is :",text)
  
```

Fig 4.02

Result after solving the expression from Image

```

In [5]: runfile('E:/Forsk/Project/college/expression/expression/test.py', wdir='E:/Forsk/
Project/college/expression/expression')
expression is : 5+5-5+9 +28-2+5
Value of expression is : 45

In [6]:
  
```

Fig 4.03

4.2 Sprint Backlog- 2

Our Sprint Backlog 2 is based on designing the UI with the help of HTML, CSS, Bootstrap, JavaScript and deploy the python code with UI can be done with the help of Flask app. In UI, we have design an input box that take file(image) as input and after taking input it displays the image in another box and after click on submit it display the result on same page. Without reloading the UI, we have to display the result, so that we can use AJAX. At last create a Text box in UI, so that we can type the expression also rather taking images every time.

Expressions (Store in png format)

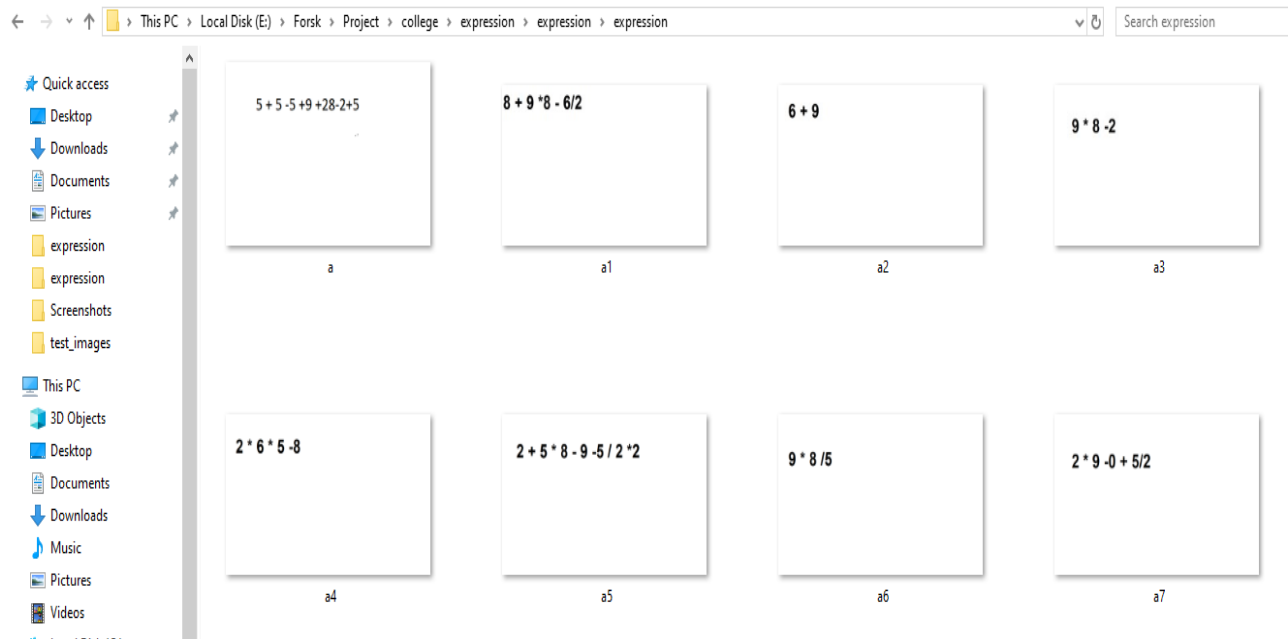


Fig 4.04

UI for our Project

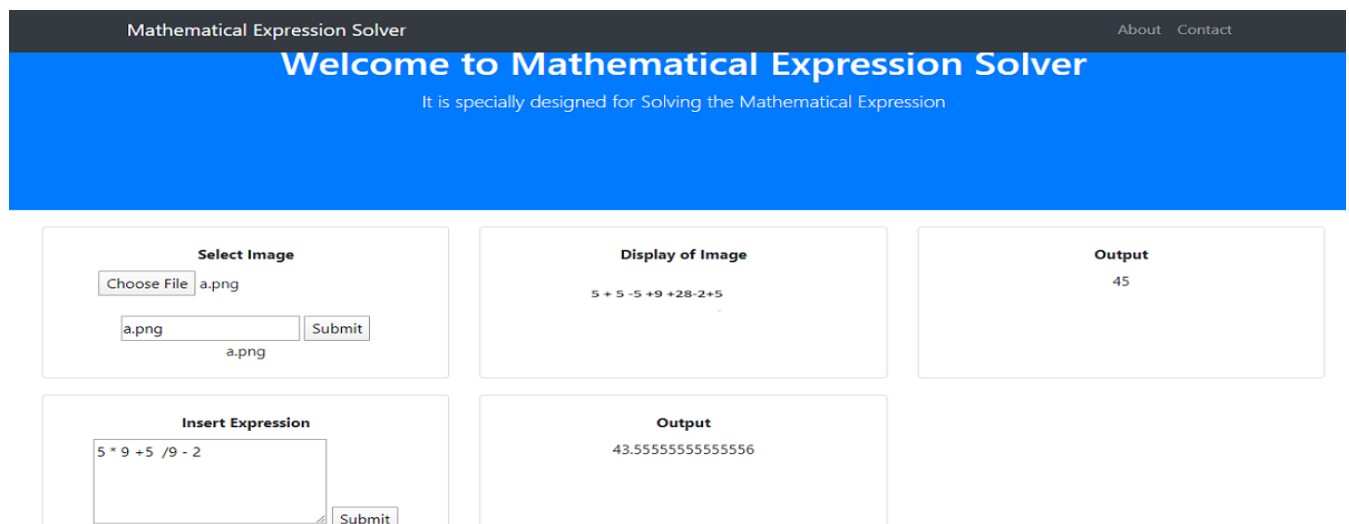
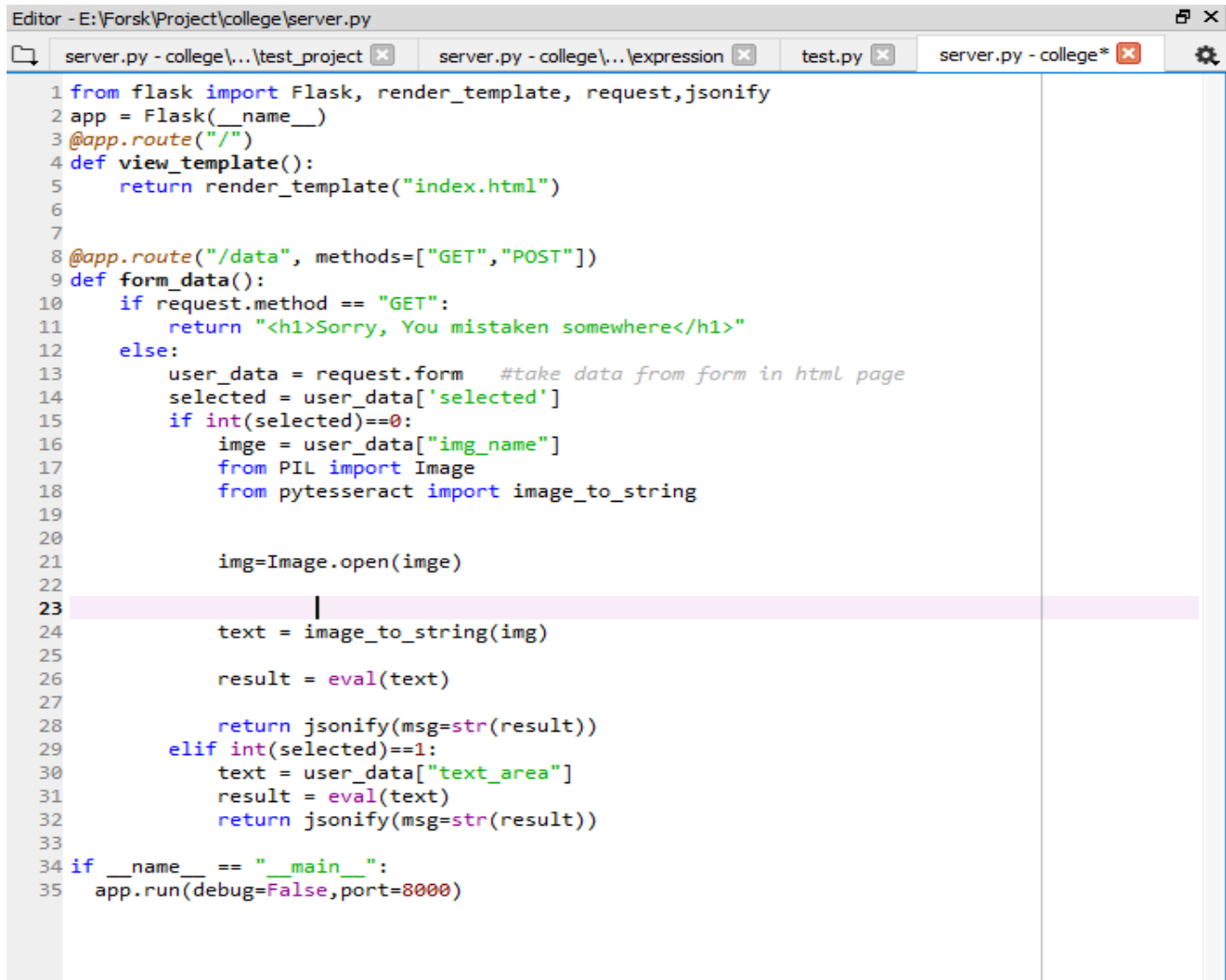


Fig 4.05

Sample Code for UI connect with flask



```

1 from flask import Flask, render_template, request, jsonify
2 app = Flask(__name__)
3 @app.route("/")
4 def view_template():
5     return render_template("index.html")
6
7
8 @app.route("/data", methods=["GET", "POST"])
9 def form_data():
10     if request.method == "GET":
11         return "<h1>Sorry, You mistaken somewhere</h1>"
12     else:
13         user_data = request.form #take data from form in html page
14         selected = user_data['selected']
15         if int(selected)==0:
16             image = user_data["img_name"]
17             from PIL import Image
18             from pytesseract import image_to_string
19
20
21             img=Image.open(image)
22
23             |
24             text = image_to_string(img)
25
26             result = eval(text)
27
28             return jsonify(msg=str(result))
29         elif int(selected)==1:
30             text = user_data["text_area"]
31             result = eval(text)
32             return jsonify(msg=str(result))
33
34 if __name__ == "__main__":
35     app.run(debug=False,port=8000)

```

Fig 4.06

4.3 Sprint Backlog- 3

Our Sprint Backlog 3 is upgradation on the previous sprint i.e. In previous sprint we solve only simple expression but now we are going to work on finding the single root or multiple roots of expression and try to solve various types of differentiation & integration. All these types of equations are solved either with the help of image (extracting the expression from image) or we can write these expressions in the text box.

By, completing Sprint backlog 3, we have almost complete the maximum work related to expression that can be solved i.e. solving the normal expression of addition, subtraction to various quadratic equations, differentiation, integration etc.

Sample Code of UI Integrate with Flask

```

Editor - E:\Forsk\Project\college\expression\expression\server.py
server.py - test_project x server.py - expression* x test.py x

1 from flask import Flask, render_template, request, jsonify
2 app = Flask(__name__)
3 import pickle
4 with open('solution_evaluator.pkl', 'rb') as f1:
5     solution_evaluator = pickle.load(f1)
6 with open('Equation_creator.pkl', 'rb') as f1:
7     Equation_creator = pickle.load(f1)
8 with open('variable.pkl', 'rb') as f1:
9     variable = pickle.load(f1)
10 from sympy import *
11 q,w,e,r,t,y,u,i,o,p = variable("q w e r t y u i o p")
12 a,s,d,f,g,h,j,k,l = variable("a s d f g h j k l")
13 z,x,c,v,b,n,m = variable("z x c v b n m")
14 Q,W,E,R,T,Y,U,I,O,P = variable("Q W E R T Y U I O P")
15 A,Q,S,D,F,G,H,J,K,L = variable("A Q S D F G H J K L")
16 Z,X,C,V,B,N,M = variable("Z X C V B N M")
17 @app.route("/")
18 def view_template():
19     return render_template("index.html")
20 @app.route("/data", methods=["GET", "POST"])
21 def form_data():
22     if request.method == "GET":
23         return "<h1>Sorry, You mistaken somewhere</h1>"
24     else:
25         user_data = request.form
26         selected = user_data['selected']
27         if int(selected)==0:
28             img = user_data["img_name"]
29             import pytesseract
30             pytesseract.pytesseract.tesseract_cmd = 'C:\\\\Program Files (x86)\\Tesseract-OCR\\tesseract.exe'
31             from PIL import Image
32             image = Image.open(img)
33             text = pytesseract.image_to_string(image)
34             print(text)
35             text = text.strip()
36             text = text.replace("^", "***")

```

Fig 4.07

```

Editor - E:\Forsk\Project\college\expression\expression\server.py
server.py - test_project x server.py - expression* x test.py x

37 try:
38     result = eval(text)
39 except:
40     try:
41         variables = {}
42         var = 'qwertyuiopasdfghjklzxcvbnmQWERTYUIOPASDFGHJKLZXCVBNM'
43         equations = text.split("\n")
44         left = []
45         right = []
46         for eq in equations:
47             a = eq.index('=')
48             print(a)
49             left.append(eq[:a])
50             right.append(eq[a+1:])
51         for char in text:
52             if char in var:
53                 if variables.get(char):
54                     variables[char] += 1
55                 variables[char] = 1
56         q,w,e,r,t,y,u,i,o,p = variable("q w e r t y u i o p")
57         a,s,d,f,g,h,j,k,l = variable("a s d f g h j k l")
58         z,x,c,v,b,n,m = variable("z x c v b n m")
59         Q,W,E,R,T,Y,U,I,O,P = variable("Q W E R T Y U I O P")
60         A,Q,S,D,F,G,H,J,K,L = variable("A Q S D F G H J K L")
61         Z,X,C,V,B,N,M = variable("Z X C V B N M")
62         query = "solution_evaluator(["
63         for i in range(len(equations)):
64             query = query + 'Equation_creator('+str(left[i])+',' +str(right[i])+'),'
65             query = query[:-1] + '], ['
66             for v in variables.keys():
67                 query = query + v + ','
68             query = query[:-1] + '])'
69             result = eval(query)
70         except:
71             result = "It cannot be evaluated"
72         return jsonify(msg=str(result).replace('***', '^'))
73     elif int(selected)==1:
74         text = user_data["text_area"]

```

Fig 4.08

Expression (Store in png format)

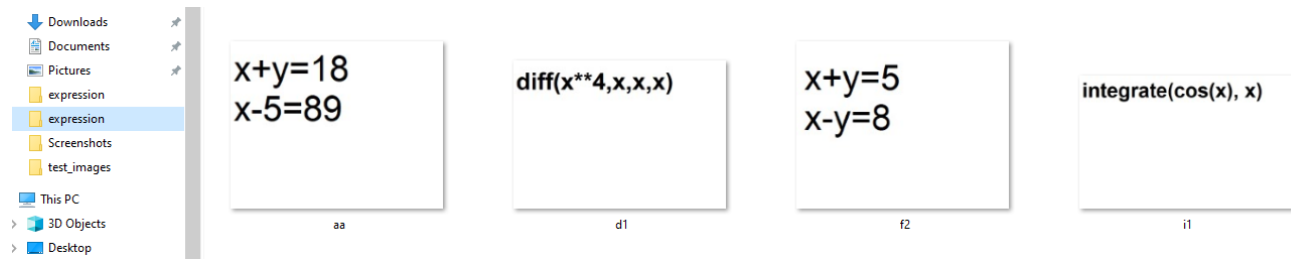


Fig 4.09

UI for our project

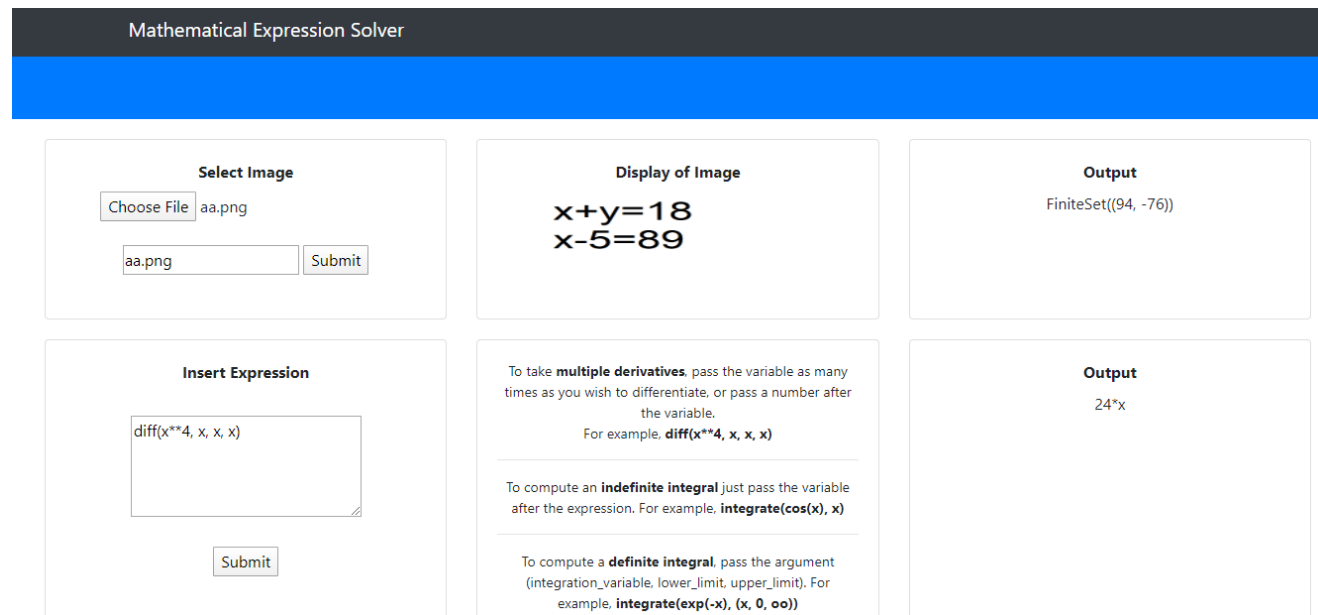


Fig 4.10

4.4 Sprint Backlog- 4

Our Sprint Backlog 4 is basically based on handwritten expression, for fetching the handwritten expression from image we have to collect the dataset of handwritten alphabet, number, operators and then train the model with the help of Convolution neural network (CNN), we select CNN model because it gives better accuracy than any other models. After model is trained we have to pass the images that contains handwritten expression then the model predicts the expression that written in image and store in new variable, after that it solves the expression and gives the result.

Handwritten Expression (Store in png format)

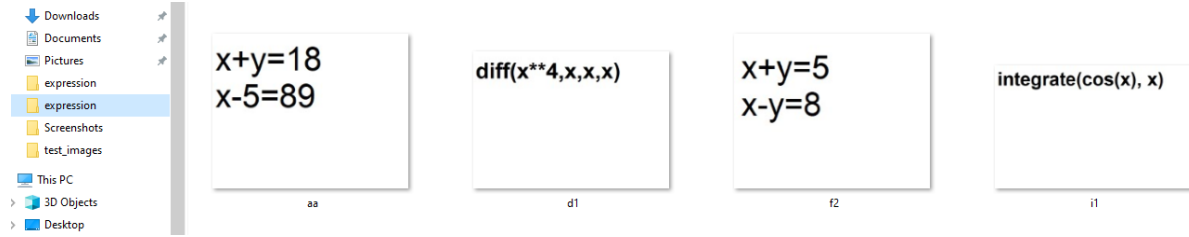


Fig 4.11

Sample code for training model for handwritten expression

```

Editor - E:\Forsk\Project\college\expression\test_project\test1.py
server.py - test_project x server.py - expression x test1.py* x
1 import cv2
2 import numpy as np
3 from keras import backend as K
4 K.common.set_image_dim_ordering('th')
5 from keras.models import model_from_json
6 def solve(img1):
7     K.clear_session()
8     global batch_index
9     json_file = open('pickles/john.json', 'r')
10    loaded_model_json = json_file.read()
11    json_file.close()
12    loaded_model = model_from_json(loaded_model_json)
13    # Load weights into new model
14    loaded_model.load_weights("pickles/john_final.h5")
15    img1="test_images/"+img1
16    img = cv2.imread(img1,cv2.IMREAD_GRAYSCALE)
17    #erosion = cv2.erode(img,kernel,iterations = 3)
18    #dilation = cv2.dilate(img,kernel,iterations = 1)
19    #img=dilation
20    if img is not None:
21        #images.append(img)
22        img=~img
23        ret,thresh=cv2.threshold(img,127,255,cv2.THRESH_BINARY)
24        ctrs,ret=cv2.findContours(thresh,cv2.RETR_TREE,cv2.CHAIN_APPROX_SIMPLE)
25        cnt=sorted(ctrs, key=lambda ctr: cv2.boundingRect(ctr)[0])
26        w=int(28)
27        h=int(28)
28        train_data=[]
29        #print(len(cnt))
30        rects=[]
31        for c in cnt :
32            x,y,w,h= cv2.boundingRect(c)
33            rect=[x,y,w,h]
34            rects.append(rect)
35        #print(rects)

```

Fig 4.12

Sample Flask app (handwritten)

```

Editor - E:\Forsk\Project\college\expression\test_project\server.py
server.py - test_project*  server.py - expression  test.py
1 from flask import Flask, render_template, request, jsonify, url_for
2 from test1 import solve
3 app = Flask(__name__)
4 @app.route("/")
5 def view_template():
6     return render_template("index.html")
7 @app.route("/data", methods=["GET", "POST"])
8 def form_data():
9     if request.method == "GET":
10        return "<h1>Sorry, You mistaken somewhere</h1>"
11    else:
12        user_data = request.form #take data from form in html page
13        selected = user_data['selected']
14        if int(selected)==0:
15            imge = solve(user_data["img_name"])
16            print(imge)
17            result=imge[1]
18            return jsonify(msg=str(result))
19 if __name__ == "__main__":
20     app.run(debug=False, port=8000)

```

Fig 4.13

UI for Handwritten Expression Solver

Mathematical Expression Solver

About Contact

Welcome to Mathematical Expression Solver

It is specially designed for Solving the Mathematical Expression

Select Image

new11.png

Display of Image

$$100 + 2100 \times 500$$

Output

1050100

Copyright © Mathematical Expression Solver 2019

Fig 4.14

Chapter 5: Result & Future Scope

Mathematical expression solver can evaluate the expressions which are either printable text or handwritten expression without any error and the expression can be a simple expression or it can be an expression with the variable (linear, quadratic, etc.) or it can be the expression with the keyword diff and integrate where integrate is used for the expression in which we have to find out the integration of the expression. We can also set the limits to it that means we can find the integration which has limits. diff is used where we have to find out the differentiation of the expression.

5.1 Future Scope:

Now we are getting the images in the form of images but in future, we will use the webcam to get the algebraic expression. Whatever we are writing in front of webcam our project will evaluate the solution. Our project will also able to find the value of the independent variable. This will also able to solve the quadratic expression and ability to write the multiple values of undefined variables write now it is doing that for printable text only in future we will try to do the same for the handwritten expressions. This project will also able to do the comparison operation and will return the result in the form of Boolean values like true or false write now it is doing that for printable text only in future we will try to do the same for the handwritten expressions. We try to make this project is useful for the learning platforms, so that for matching the result of expression from their own solution due to their learners from their platforms saves their time and help them to get knowledge fastly. This platform is also useful for solving the large complex problems easily in the live platform.

5.2 RESULT:

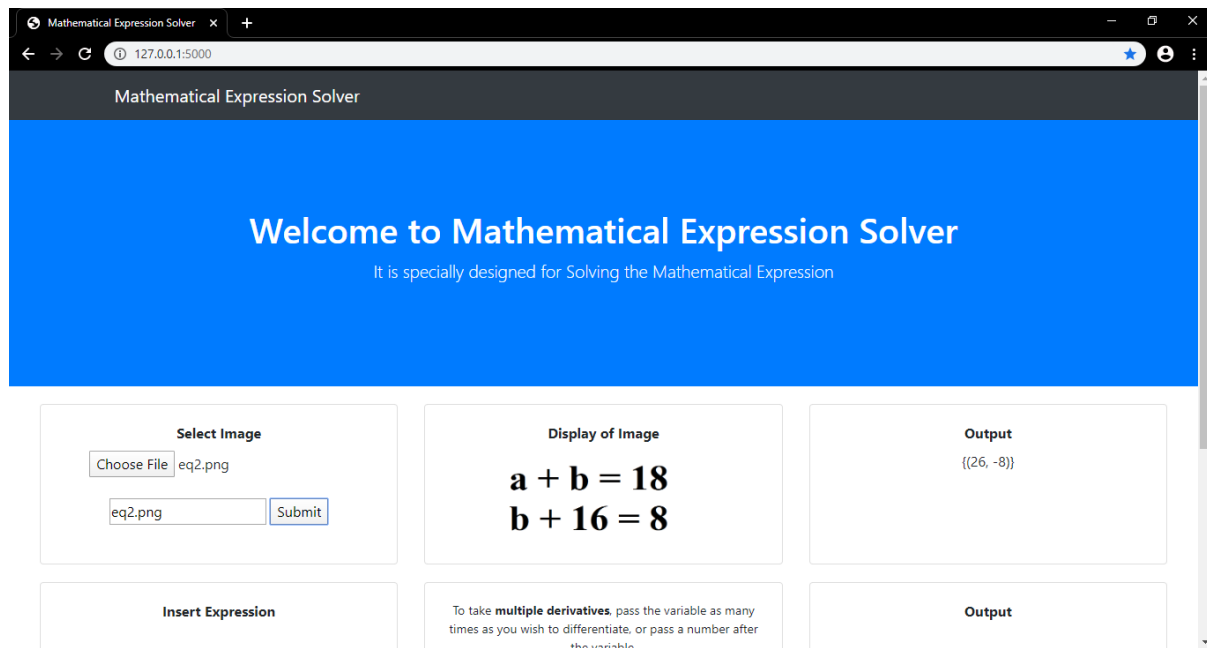


Fig 5.01

In the above example, we take the image of an expression which is written in the printable text format and then after fetching the expression our project applies the optimal algorithm to find out the result of the expression and after getting the result it is shown on the interface using flask.

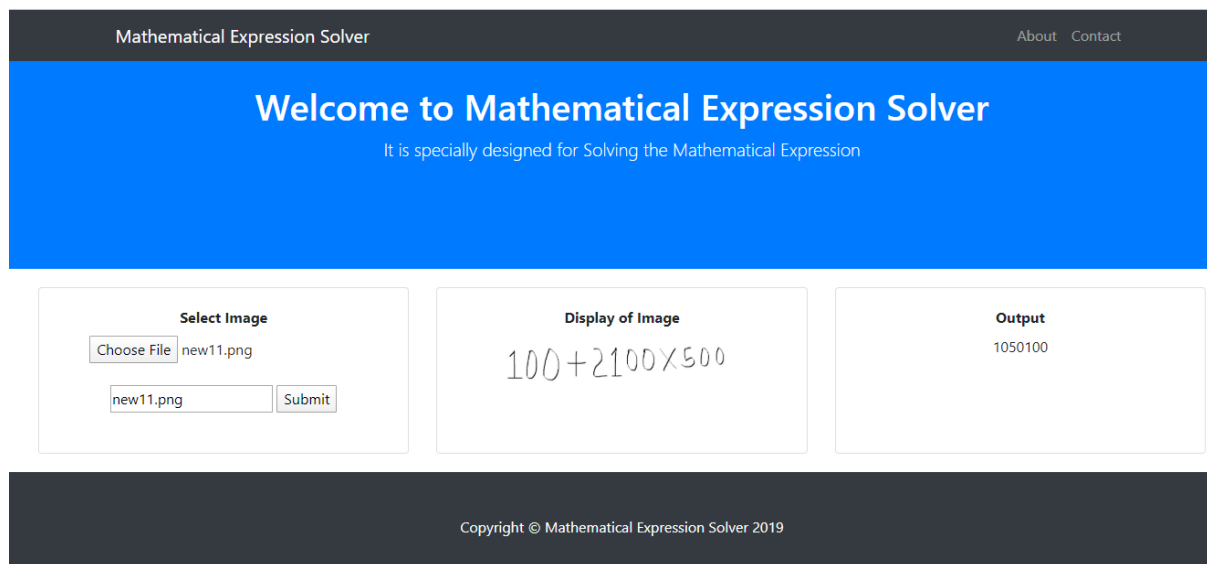


Fig 5.02

In the above example, we take the image of an expression which is written in the handwritten text format and then after fetching the expression our project applies the optimal algorithm to find out the result of the expression and after getting the result it is shown on the interface using flask.

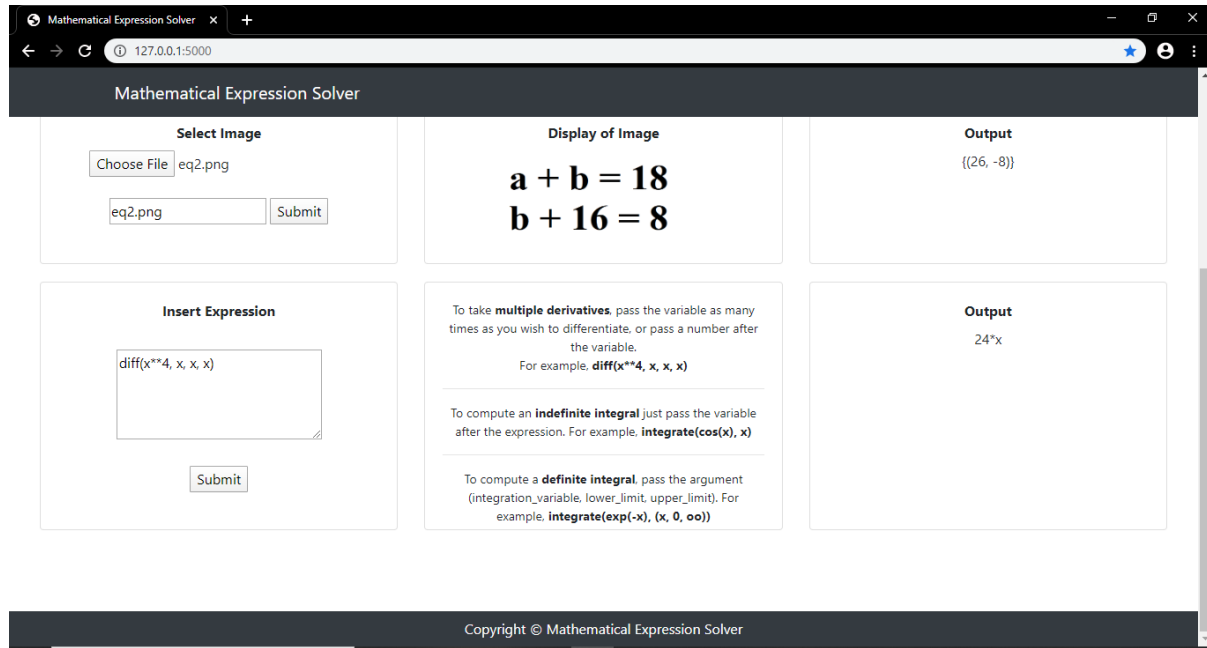


Fig 5.03

In the above example, we write an expression using keyword diff which is used to evaluate the differentiation after fetching the expression our project applies the optimal algorithm to find out the result of the expression and after getting the result it is shown on the interface using flask.

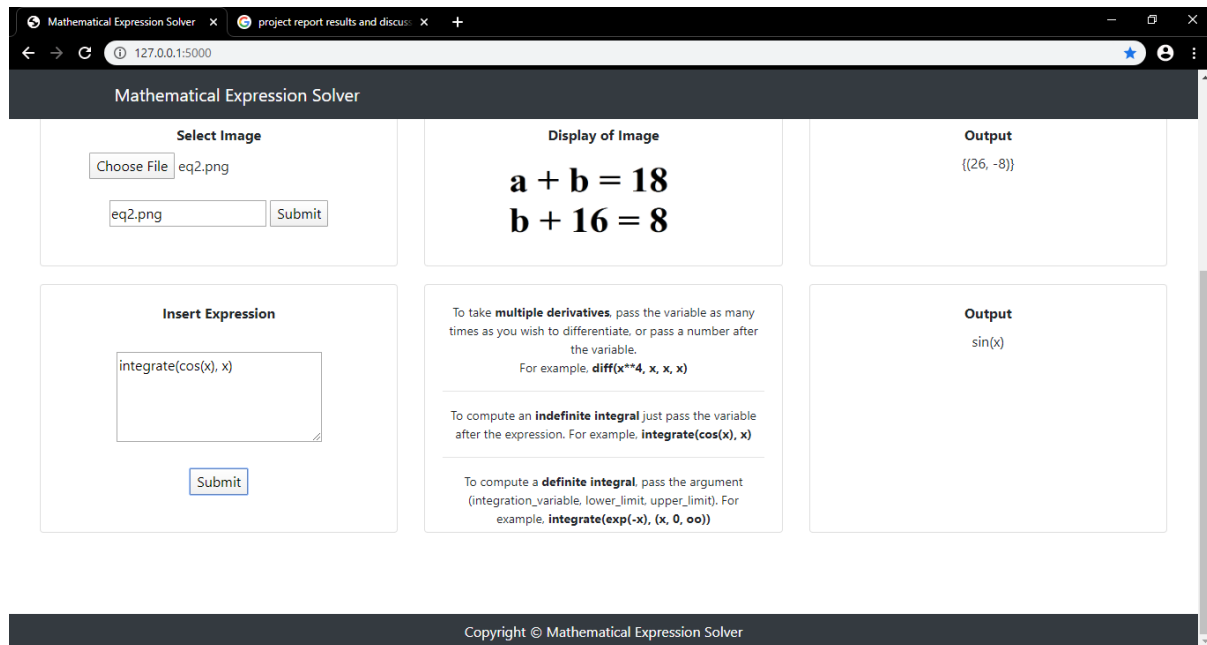


Fig 5.04

In the above example, we write an expression using keyword integrate which is used to evaluate the integration after fetching the expression our project applies the optimal algorithm to find out the result of the expression and after getting the result it is shown on the interface using flask.

5.3 Working Architecture (of Handwritten Expression Solver)

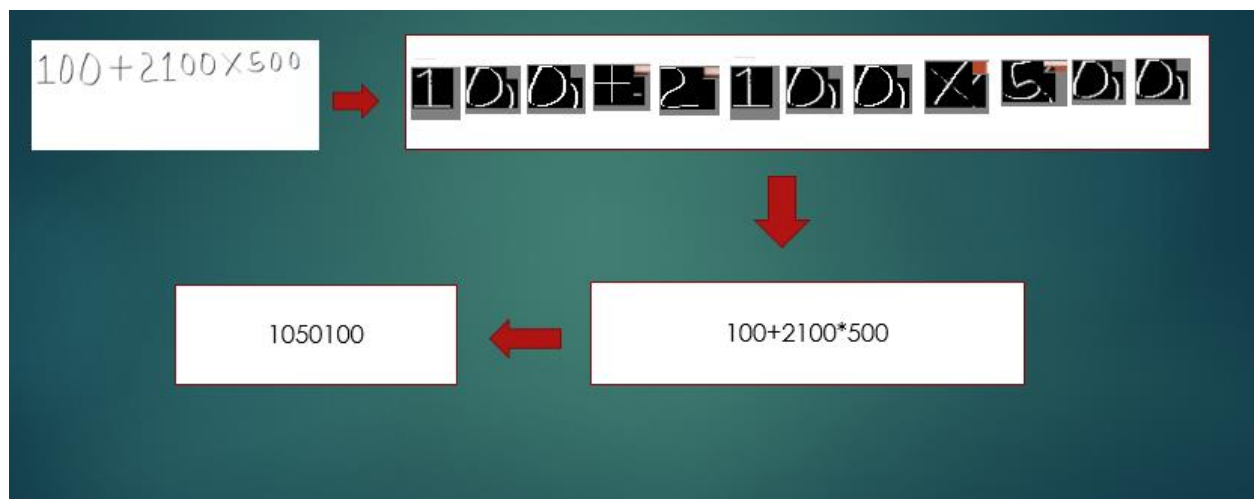


Fig 5.05

Chapter 6: Limitations

There are certain limitations of this research as anything cannot be perfect and there is always a darker side which are to be explored. With this context, some limitations are listed below –

- There is problem in fetching divide sign i.e. “/” from handwritten text because most of time it read divide as “1”.
- If the image of handwritten character contains noise or if the image is blur, then it is very difficult to detect the handwritten character from that image.
- Most of times we write multiply as “x” in handwritten format but system read multiply as “*”, so we have to detect “x” and convert it into “*”.
- Sometimes we also observe that while writing “+” in handwritten character, it may be read as “x” because in handwritten sometime “+” will be tilted and read as “x”.
- There is difficulty in identifying characters like Zero “0” or alphabet “O”, one “1” or alphabet “l” or alphabet “i” or alphabet “l”.
- There is also difficulty in identifying character like multiply “x” or alphabet “x” because both are similar.
- If there are certain watermark in image, then it is very difficult to fetch the expression from the image.
- In some case character like “8” read as alphabet “O” or alphabet “o” or digit “0”.

REFERENCES

1. Pratik Madhukar Manwatkar, Dr. Kavita R. Singh "A Technical Review on Text Recognition from Images" in 9th International Conference on Intelligent Systems and Control (ISCO) ,2015.
2. Savita Choudhary, Nikhil Kumar Singh and Sanjay Chichadwani "Text Detection and Recognition from Scene Images using MSER and CNN" in Second International Conference on Advances in Electronics, Computer and Communications (ICAECC), 2018.
3. Tan Chiang Wei, U. U. Sheikh, Ab Al-Hadi Ab Rahman "Improved Optical Character Recognition with Deep Neural Network" in IEEE 14th International Colloquium on Signal Processing & its applications (CSPA), 2018.
4. Rohan Vaidya, Darshan Trivedi, Sagar Satra, Mrunalini Pimpale "Handwritten Character Recognition Using DeepLearning " in 2nd International Conference on Inventive Communication and Computational Technologies (ICICCT), 2018.
5. J. Pradeepa, E. Srinivasana, S. Himavathib, "Neural Network Based Recognition System Integrating Feature Extraction and Classification for English Handwritten," International journal of Engineering, Volume 25, May 2012.