Optomath

Image Equation Solver

Varun Nare
Department of Electronics & Telecommunication
Rizvi College of Engineering
Mumbai, India
varunnare@eng.rizvi.edu.in

Kshitij Surve
Department of Electronics & Telecommunication
Rizvi College of Engineering
Mumbai, India
kshitijsurve1999@eng.rizvi.edu.in

Haider Daresalamwala
Department of Electronics & Telecommunication
Rizvi College of Engineering
Mumbai, India
haiderdares@eng.rivi.edu.in

Bapu Sutar
Department of Electronics & Telecommunication
Rizvi College of engineering
Mumbai, India
bapusutar92@eng.rizvi.edu.in

Prof. R.S. Deshmukh
Department of Electronics
Rizvi College of Engineering
Mumbai, India
rajansdes11@gmail.com

Prof. Junaid Mandviwala
Department of Electronics & Telecommunication
Rizvi College of Engineering
Mumbai, India
junaid@eng.rizvi.edu.in

***Abstract*— Scientific calculators can be quite tedious to use. Solving a problem requires multiple inputs which can be unintuitive and intimidating for new users. Optomath eliminates all these problems by streamlining the process. The user needs to provide two inputs, the mode of operation and the image. This method not only saves time but also removes the element of human error which is involved in the traditional process.**

The objective of this paper is to summarize the working of our application and its limitations. This paper serves the purpose of presenting state of the art results and techniques of and also our application provides research directions by highlighting research gaps.

Keyword- Optical Character recognization(OCR), OptoMath ,PhotoMath, Python, PyTesseract

I. INTRODUCTION

The digital era has brought in various new useful components to one's way of learning. Every student needs a quick and easy answer to every question. Thus, there's a need to improve the learning experience of the students. Optomath is such an application which is being developed to give an answer to the question that they are looking for by just clicking a photo of the math problem.

The user has to click the photo of the math problem and then select the operation which the user wants to perform which is arithmetic, linear equation or quadratic equation. Then pytesseract will read and assemble the characters of the given input into a string and then the application while perform the operation selected by the user.

Pytesseract plays a vital role in here as it is of utmost importance to extract the images with high level of accuracy and give the correct answer.

Pytesseract is an optical character recognition tool for python. That is, it will recognize and "scan" the text embedded in images. It can also be used 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, and others. Additionally, if used as a script, Pytesseract will print the recognized text instead of writing it to a file.[1]

This application while help the students to cross check their solutions whether they have solved the problem correctly or not.

II. SYSTEM OVERVIEW AND WORKING

Optomath is a software that converts input image that contains mathematical equations into a valid raw string literal and then precise algorithms convert the raw text into valid equations which are stored in desired data structures and then we pass these normalized equations into their respective solvers to get the result. We have made three operational functions with their valid extractors and solvers the three functions are arithmetic equation solver, linear equation solver and quadratic equation solver.

A. Block diagram

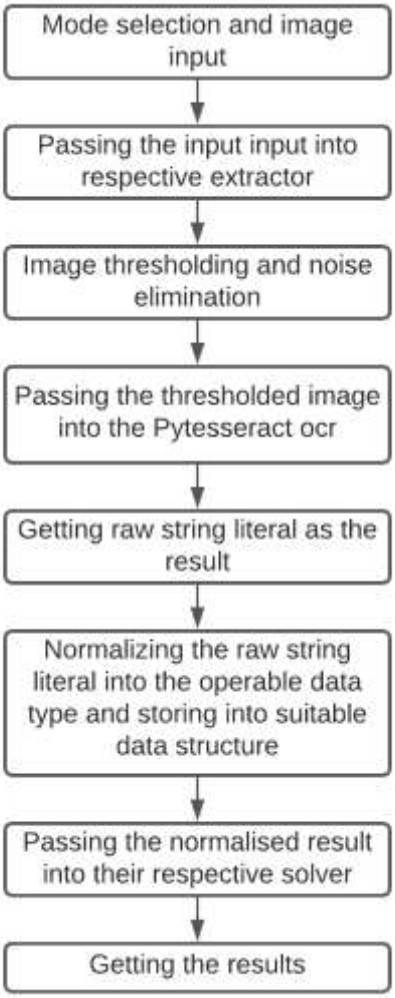

Fig.1

Fig1: Flow Chart of OptoMath

1. Mode selection: -The very first step is to select the mode in which the Optomath calculator should work on so there are three mode in which it can work
- arithmetic solver
- linear equation solver
- quadratic equation solver

2. Image input and data extraction using Tesseract OCR: -

After selecting the mode of our calculator will pass the image as an input to the respective extractor which are invoked accordingly to selected mode and each mode has its respective extractor and solver

So, when we pass the image to the extractor it will remove noise from the image and threshold the image. This is done so that if the image is noisy then the tesseract's accuracy in detecting the data is reduced.[2]

Tesseract is an optical character recognition engine for various operating systems. It is free software, released under the Apache License [3]. Originally developed by HP as proprietary software in the 80s, it was released as open source in 2005 and development has been sponsored by Google since 2006

After extracting the data form the pytesseract we get the data as a raw string literal now the extractor will return the data and we pass this raw string literal into the respective solvers

3. Solvers: -

Now as data which we have extracted is in raw sting format we need to extract the operable data from that raw string and store them in operable data type and then into respective data structures to do that we have implement normalize functions and the solvers call this normalize functions as a sub route to get the desired result and each solver has a different normalize functions

Now this normalizes data which is returned by the normalize methods are then passed to the respective helper solver functions which actually gets the result for us

Now in the Linear Equation solver this helper solver function is the gaussian elimination algorithm [4]. In the quadratic equation solver, the helper solver function is the quadratic formula. In the arithmetic equation solver, the helper solver function is the algorithm designed to solve the arithmetic equation which handles the all the priorities and hierarchal structure of our arithmetic equations (BODMAS) and this helper solver methods are also called as subroutine by our solver function and then finally the result is passed to the result print function which actually prints and formats the results

Then the result is displayed and the program is terminated.

III. RESULTS

- Linear equation solver

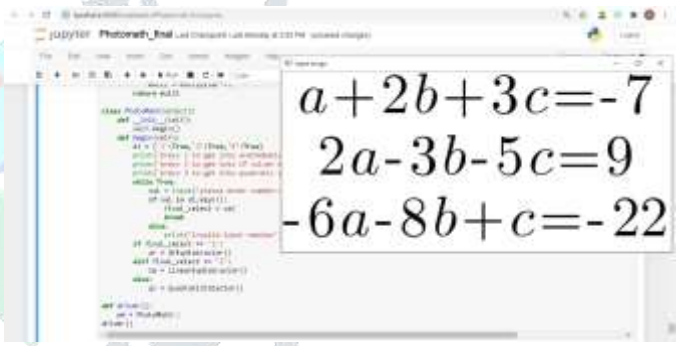

Fig 2

Fig 2: - Input Image

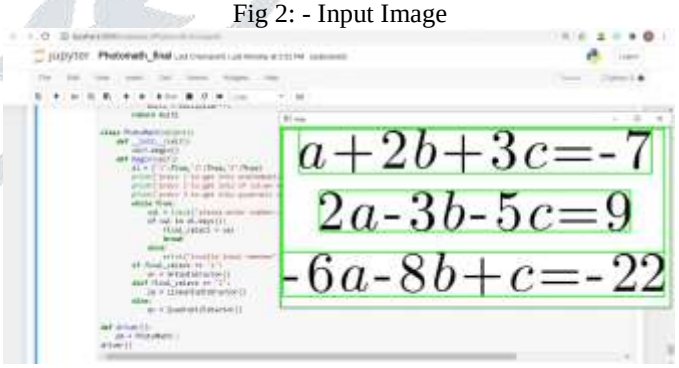

Fig 3

Fig 3: - Data Detection and Extraction

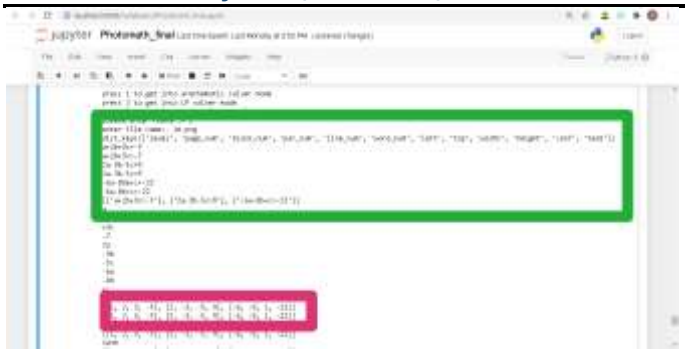

Fig 4

Fig 4: - Extraction Normalization

Fig 5

Fig 5: - Helper Solver (Gaussian Elimination) and Result

- Arithmetic equation solver

Fig 6

Fig 6: - Input image

Fig 7

Fig 7: - Arithmetic Equation detection

Fig 8

Fig 8: - Data extraction and normalization

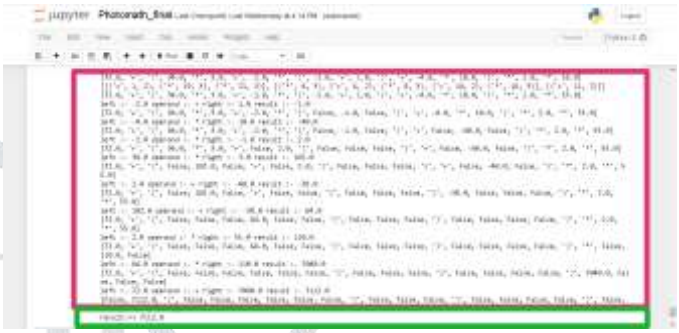

Fig 9

Fig 9:- Helper Solver and Result

- Quadratic Equation

Fig 10

Fig 10: - Quadratic equation input

Fig 11

Fig 11: - Equation detection

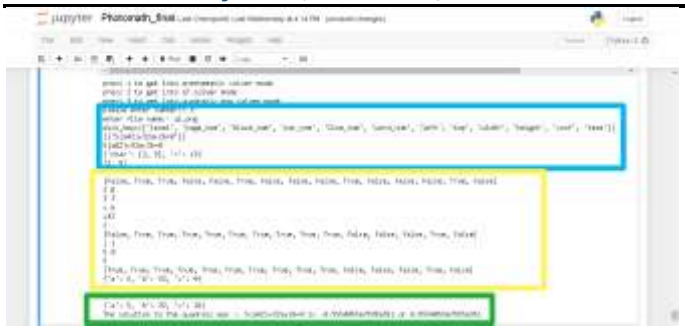

Fig 12

Fig 12: - Data extraction, normalization, helper solver and result

IV. CONCLUSION

All the three modes namely arithmetic, quadratic and linear equations were successfully performed with accurate results. We used pytesseract to detect and extract equations from images. We used type specific algorithms to normalize the raw inputs and to get the results. Some of the limitations of our project is that it is totally reliant on the accuracy of pytesseract, as it is not 100% accurate. In some cases, we get incorrect output. Let's say we provide an image of matrix the tesseract will consider entire row of the matrix to be a single element hence it is not possible to parse matrix by using tesseract. We are working on another way of extracting data from the image of matrix by box bound method using k-nearest-neighbors [5] this method is able to extract elements of the matrix easily.

V. FUTURE ENHANCEMENTS

Since the present software is limited to solving only the arithmetic, quadratic equations and linear equations there can be introduction of new operations as well. For example, matrixes, plotting of graph through equations and word problems.

The software can be made much more interactive, this will promote an excellent intuitive learning environment for the students.

We will be working on making a deep learning model [6] which will be added as another layer after taking the inputs, which will detect the mode to be implemented without requiring additional input from the client's side.

ACKNOWLEDGMENT

The authors are grateful to Rizvi College of Engineering, India for providing the necessary facilities and guidance to carry out this project.

REFERENCES

[1] Akash V Pavaskar, Akshay S Accha, Anoop R Desai, Darshan K L . "INFORMATION EXTRACTION FROM IMAGES USING PYTESSERACT AND NLTK" , May 2017, Volume 4, Issue 05, JETIR (ISSN-2349-5162).

[2] Sahu, Narendra & Sonkusare, Manoj. (2017). A Study on Optical Character Recognition Techniques. International Journal of Computational Science, Information Technology and Control Engineering. 4. 01-15. 10.5121/ijcsitce.2017.4101. R. Nicole, "Title of paper with only first word capitalized," J. Name Stand. Abbrev., in press.

[3] G. Abdul Robby, Antonia Tandra, Imelda Susanto, Jeklin Harefa, Andry Chowanda,Implementation of Optical Character Recognition using Tesseract with the Javanese Script Target in Android Application,Procedia Computer Science,Volume 157,2019,Pages 499-505,

[4] Joseph F. Grcar,How ordinary elimination became Gaussian elimination,Historia Mathematica,Volume 38, Issue 2,2011,Pages 163-218,

[5] Zhang Z. Introduction to machine learning: k-nearest neighbors. *Ann Transl Med*. 2016;4(11):218. doi:10.21037/atm.2016.03.37

[6] I. H. Kazi and D. J. Lilja, "Coarse-grained thread pipelining: a speculative parallel execution model for shared-memory multiprocessors," in *IEEE Transactions on Parallel and Distributed Systems*, vol. 12, no. 9, pp. 952-966, Sept. 2001, doi: 10.1109/71.954629