

HANDWRITTEN MATH SOLVER

Tribhuvan University
Institute of Science and Technology



A FINAL YEAR PROJECT SUBMISSION IN
PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE
DEGREE OF BACHELORS OF SCIENCE IN COMPUTER SCIENCE
AND INFORMATION TECHNOLOGY

UNDER THE SUPERVISION OF

Mr. Tek Raj Joshi

SUBMITTED BY

Raj Bhattarai (21164/075)

Rasmita Gautam (21165/075)

Sanju Kc (21174/075)

SUBMITTED TO

TRINITY INTERNATIONAL COLLEGE

Department of Computer Science and Information Technology

Dillibazar Height, Kathmandu, Nepal

12 May 2023

HANDWRITTEN MATH SOLVER

Tribhuvan University
Institute of Science and Technology



A FINAL YEAR PROJECT SUBMISSION IN
PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE
DEGREE OF BACHELORS OF SCIENCE IN COMPUTER SCIENCE
AND INFORMATION TECHNOLOGY

SUBMITTED BY

Raj Bhattarai (21164/075)
Rasmita Gautam (21165/075)
Sanju Kc (21174/075)

SUBMITTED TO

TRINITY INTERNATIONAL COLLEGE
Department of Computer Science and Information Technology
Dillibazar Height, Kathmandu, Nepal

12 May 2023

DECLARATION

The project entitled “**Handwritten Math Solver**” is being submitted to the Department of Computer Science and Information Technology, Dillibazar, Kathmandu, Nepal for the fulfillment of the final year project under the supervision of **Mr. Tek Raj Joshi**.

This project is original and has not been submitted earlier in part or full in this or any other form to any university or institute, here or elsewhere, for the award of any degree.

.....

Raj Bhattarai (21164/075)

Rasmita Gautam (21165/075)

Sanju Kc (21174/075)

Date: 12 May 2023

RECOMMENDATION

This is to recommend that **Raj Bhattarai, Rasmita Gautam** and **Sanju Kc** has carried out research entitled “**Handwritten Math Solver**” for the fulfillment of final year project in Bachelor of Science in Computer Science and Information Technology under my supervision. To my knowledge, this work has not been submitted for any other degree.

They have fulfilled all the requirements laid down by the Trinity International College Department of Computer Science and Information Technology, Dillibazar, Kathmandu.

.....

Mr. Tek Raj Joshi

Project Supervisor

Department of Computer Science and Information Technology

Trinity International College

Dillibazar Height, Kathmandu

Date: 12 May 2023



Dillibazar Height, PO Box: 26111, Kathmandu,

NepalTel: +977 1 4445955/4445956, Fax:

4437867

Email: info@trinitycollege.edu.np

www.trinitycollege.edu.np

LETTER OF APPROVAL

Date: 12 May 2023

On the recommendation of Mr. Tek Raj Joshi, this Project Report submitted by **Raj Bhattarai, Rasmita Gautam and Sanju KC** entitled “**Handwritten Math Solver**” in partial fulfillment of the requirement for the award of the bachelor’s degree in Computer Science and Information Technology is a bonafide record of the work carried out under my guidance and supervision at Trinity International College, Dillibazar, Kathmandu.

EVALUATION COMMITTEE

.....
Satya Bahadur Maharjan
Program Coordinator,
Department of Computer Science and
Information Technology,
Trinity International College
Dillibazar Height, Kathmandu, Nepal

.....
Tek Raj Joshi
Project Supervisor
Trinity International College

.....
External

Date:

ACKNOWLEDGEMENT

We deeply appreciate Trinity International College for providing us with the opportunity to work on a project as part of our syllabus. Additionally, we express our gratitude to our Coordinator Mr. Satya Bahadur Maharjan, and Assistant Coordinator Mr. Abhishek Dewan, for guiding us throughout the project and motivating us to put in extra effort.

We are also indebted to our supervisor, Mr. Tek Raj Joshi, for his continuous help and direction, particularly his significant recommendations and suggestions that helped us overcome the challenges we faced during the project's development.

We would like to extend our sincere thanks to our esteemed principal, and all our respected educators, whose encouragement and motivation paved the way for the successful completion of this project. We are also grateful to all the faculty members who directly or indirectly assisted us in completing this project. Lastly, we would like to thank our friends, seniors, and colleagues for their valuable suggestions, comments, and support.

Thank you!

Raj Bhattarai (21164/075)

Rasmita Gautam (21165/075)

Sanju K.C (21174/075)

ABSTRACT

This report shows how we can implement an algorithm for the detection of handwritten expression evaluators to build a system that will evaluate the results of the raw user input or image. As we know, we have to manually calculate handwritten expressions which are tedious and time-consuming. In the traditional method, the user has to use a calculator or calculate the result of the expression themselves. This proposed solution is to develop a working prototype of a system that provides results based on the handwritten digits and gives the value of the expression. The main goal of this project is to use the Convolutional Neural Network (CNN) algorithm to identify the handwritten number. Another part of this system is to give the result automatically based on the recognized digit and expression.

Keywords: *Digit Recognition, Convolution Neural Network*

Table Of Contents

DECLARATION	iii
RECOMMENDATION	iv
LETTER OF APPROVAL.....	v
ACKNOWLEDGEMENT	vi
ABSTRACT.....	vii
LIST OF FIGURES	x
LIST OF ABBREVIATION	xi
LIST OF TABLES	xii
CHAPTER 1.....	1
INTRODUCTION.....	1
1.1. Introduction.....	1
1.2. Problem statement	1
1.3. Objective	2
1.4 Scope and limitations	2
1.5. Development Methodology	2
1.6. Report Organization	3
CHAPTER 2.....	4
BACKGROUND STUDY AND LITERATURE REVIEW	4
2.1 Background Study:	4
2.2 Literature Review:	5
CHAPTER 3.....	7
SYSTEM ANALYSIS.....	7
3.1 System Analysis.....	7
3.1.1 Requirement Analysis.....	7
3.1.2 Feasibility Study	9
3.1.3 System Requirements:.....	11
CHAPTER 4.....	12
System Architecture.....	12
4.1 Design.....	12
4.1.1 System Architecture	12
4.1.2 FlowChart Diagram.....	13
4.1.3 Sequence Diagram	14
4.1.4 Activity Diagram.....	15
4.1.5 DFD Level	16
4.2 Algorithm Details	17

4.2.1 Convolutional Neural Network:	17
4.2.2 Convolutional Neural Network (CNN) Model:	18
CHAPTER 5.....	20
IMPLEMENTATION AND TESTING	20
5.1. Dataset.....	20
5.2 Image Processing and Extraction of subject element	21
5.2.1 Grayscaleing	21
5.2.2 Thresholding	22
5.2.3 Gaussian Blur	23
5.2.4 Contours	24
5.2.5 Modeling a Convolutional Neural Network to classify handwritten digits	24
5.2.6 Rectifier Linear Unit (ReLU).....	25
5.2.7 Soft plus.....	25
5.2.8 Softmax	25
5.3 Tools used.....	25
5.4 Testing.....	26
CHAPTER 6.....	28
6.1 Conclusion	28
6.2 Future Enhancement	28
REFERENCES.....	29
Appendix.....	30
Snapshots	30

LIST OF FIGURES

Figure 1: Use Case diagram.....	17
Figure 2: Nonfunctional requirement.....	18
Figure 3: Gantt chart.....	19
Figure 4: System Architecture.....	21
Figure 5: System Flowchart.....	22
Figure 6: Sequence diagram.....	23
Figure 7: Activity Diagram.....	24
Figure 8: DFD Level 0 Diagram.....	25
Figure 9: DFD Level 1 Diagram.....	26
Figure 10: Convolutional Neural Network Layers.....	27
Figure 11: CNN Model of the System.....	29
Figure 12: Dataset sample.....	30
Figure 13: Normal image.....	31
Figure 14: Grayscale image.....	32
Figure 15: Original image.....	33
Figure 16: Binary Thresholding.....	33
Figure 17: Binary Inverse Thresholding	33
Figure 18: Otsu thresholding	33
Figure 19: Otsu plus Binary Inverse Thresholding after applying Gaussian Blur	34
Figure 20: Individual Contours.....	34

LIST OF ABBREVIATION

CNN – Convolutional Neural Network

OMR – Optical Mark Recognition

OCR – Optical Character Recognition

SIANN – Space Invariant Artificial Neural Networks

RGB – Red Green Blue

ReLu – Rectifier Linear Unit

HOG – Histogram of Oriented Gradients

LIST OF TABLES

Table 1: Test Cases-----	26
---------------------------------	----

CHAPTER 1

INTRODUCTION

1.1. Introduction

The "HANDWRITTEN MATH SOLVER" is a remarkable application that leverages the power of Convolution Neural Network (CNN) technology to identify and evaluate handwritten numerical expressions. With this application, users can input their mathematical expressions by drawing them in a designated area of the interface. The software then utilizes its advanced recognition capabilities to identify the digits and expression entered by the user. The process of developing a model that can recognize handwritten digits and expressions is not an easy one. However, the use of CNN technology has made this task much simpler and more efficient. CNNs are a type of deep neural network that can recognize patterns in images, making them ideal for identifying handwritten digits and expressions.

To use this application, users can simply draw their expression in the designated area of the interface, and the software will identify the digits and expression. This can be extremely useful for individuals who have difficulty typing mathematical expressions, or for those who prefer to write them by hand. Moreover, the HANDWRITTEN MATH SOLVER application is not limited to recognizing numerical digits alone. It can also evaluate expressions entered by the user and provide solutions to mathematical problems. This makes it a versatile tool for individuals working with mathematical expressions on a regular basis.

Overall, the HANDWRITTEN MATH SOLVER application is a powerful and effective solution for recognizing and evaluating handwritten mathematical expressions. By utilizing CNN technology, it offers a highly accurate and efficient means of identifying user input, making it an essential tool for anyone working with mathematical expressions.

1.2. Problem statement

In the conventional system, all handwritten mathematical expressions are typically evaluated manually, which is a laborious and time-consuming task. Every calculation is performed by human

resources, leading to high costs, especially as the volume of calculations increases. This traditional approach is not only costly, but also prone to errors that can be detrimental to the accuracy of the results. In essence, the manual approach to mathematical calculations is inefficient, costly, and often unreliable. Therefore, there is a pressing need for a more efficient and accurate system that can automate mathematical calculations and provide a reliable solution to the problem.

1.3. Objective

- Recognize handwritten digits
- Provide automatic answer validation
- Saves time and cost

1.4 Scope and limitations

- Image-based calculators
- Mass Survey
- Billing System

Some of the general limitations are as follows:

- Can only recognize handwritten digits not alphabets
- Can only recognize bold and clearly written digits and operators

1.5. Development Methodology

Agile methodology is a flexible and iterative approach to software development, where the development process is divided into small incremental builds called sprints. Each sprint usually lasts for 2-4 weeks, and at the end of each sprint, a potentially shippable product increment is produced. The agile methodology emphasizes collaboration between team members, frequent communication, and continuous feedback.

In the context of the Handwritten Math Solver project, using Agile methodology allows us to quickly respond to changing requirements, prioritize user needs, and test and refine the system continuously. It also enables us to incorporate feedback from users and stakeholders throughout the development process, ensuring that the final product meets their needs.

1.6. Report Organization

So far, we have discussed about the overview of overall project. The rest of the report is organized as follow: Chapter 1 consist of Introduction part, problem statement and objectives. likewise, Chapter 2 consists the Background study and Literature review whereas Chapter 3 specifies the system analysis and requirement of the project and Chapter 4 describes about the design and developmental stages of our project and describes the basic process model. Followed by, chapter 5 deals with testing and implementation. Furthermore, chapter 5 shows our project result and finally chapter 6 conclude the entire project and recommends future activities.

CHAPTER 2

BACKGROUND STUDY AND LITERATURE REVIEW

2.1 Background Study:

The "Handwritten Math Solver" application is an example of an optical character recognition (OCR) system that is specifically designed to recognize and interpret handwritten mathematical expressions. OCR technology has been around for decades, but advancements in machine learning and neural networks have made it possible to develop highly accurate OCR systems for handwritten text. The use of Convolution Neural Network (CNN) is a popular approach in image recognition tasks, including OCR. CNNs can learn to recognize visual patterns in images and can be trained to recognize handwritten digits with high accuracy. The development of OCR systems for mathematical expressions presents unique challenges due to the complexity of mathematical symbols and notation. There is a wide variety of symbols and notations used in mathematical expressions, and they can vary greatly in style, size, and position. Furthermore, the arrangement and context of these symbols can also affect their meaning, making it essential for OCR systems to understand the context in which the symbols appear.

In recent years, there has been significant research on developing OCR systems for mathematical expressions, including handwritten expressions. Many of these systems use a combination of deep learning algorithms and traditional OCR techniques to achieve high accuracy. The "Handwritten Math Solver" application combines the use of CNN and OCR techniques to recognize and interpret handwritten mathematical expressions. The user interface allows users to input their expressions by drawing them on a designated area. The application then processes the input using its recognition and evaluation algorithms to provide the user with the solution to the expression.

Overall, the "Handwritten Math Solver" application is an example of the growing trend towards using advanced machine learning algorithms for OCR tasks, including mathematical expression recognition.

2.2 Literature Review:

Handwritten Math Solver are not commonly available. To achieve accurate and automated result generation, OMR (Optical Mark Recognition) and OCR (Optical Character Recognition) devices are typically employed. These devices offer the benefits of speed, efficiency, and high accuracy, making them a popular choice for automated digit recognition.

2.2.1 Optical Mark Recognition

Optical Mark Recognition (OMR) is a method of capturing human-marked data from documents such as surveys and tests. This process is commonly used for reading multiple-choice examination papers and questionnaires that feature lines or shaded areas. Traditional OMR devices work with a specialized scanner that shines a beam of light onto the paper and detects contrasting reflectivity at predetermined positions on the page to identify marked areas. Some OMR devices use preprinted forms on "trans optic" paper that measure the amount of light passing through the paper, allowing marks on either side of the paper to be detected.

Desktop OMR software is also available, which allows users to create forms using a word processor and print them on a laser printer. These forms can then be processed using a common desktop image scanner with a document feeder. Overall, OMR technology provides a fast and efficient way to capture and process data from documents. [1]

2.2.2 Optical Character Recognition

OCR, also known as optical character recognition, is the electronic or mechanical conversion of images of typed, handwritten, or printed text into machine-encoded text from scanned documents, photos of documents, scene photos (such as the text on signs and billboards in a landscape photo), or from subtitle text superimposed on an image (from a television broadcast, for example).

It is a common practice to digitize printed texts so they can be electronically edited, searched, stored more compactly, displayed online, and used in machine processes like cognitive computing, machine translation, and (extracted) text-to-speech. This method of digitizing printed texts is used to enter data from printed paper data records such as passports, invoices, bank statements, computerized receipts, business cards, mail, printouts of static-data, or any suitable documentation. Pattern recognition, artificial intelligence, and computer vision are all areas of study in OCR. [2]

CHAPTER 3

SYSTEM ANALYSIS

3.1 System Analysis

3.1.1 Requirement Analysis

Requirement analysis is a critical step in the software development life cycle (SDLC). During this stage, the focus is on gathering both functional and non-functional requirements for the product and assessing their feasibility. It is important to gather requirements early on in the project to establish goals and objectives. A thorough understanding of the requirements is key to the successful development of the software. Without this step, hard work alone will not lead to the desired end product. The requirement analysis stage is crucial because without a clear understanding of the exact requirements, the final output cannot be achieved as desired.

I. Functional Requirements

A functional requirement outlines the services the system must offer. How the system responds to specific inputs and how the system ought to function in specific circumstances. The following are the functional requirements for our system:

- It should detect the handwritten digit of the expression
- It should detect the operator written in the expression
- It gives the result of the expression written by the user

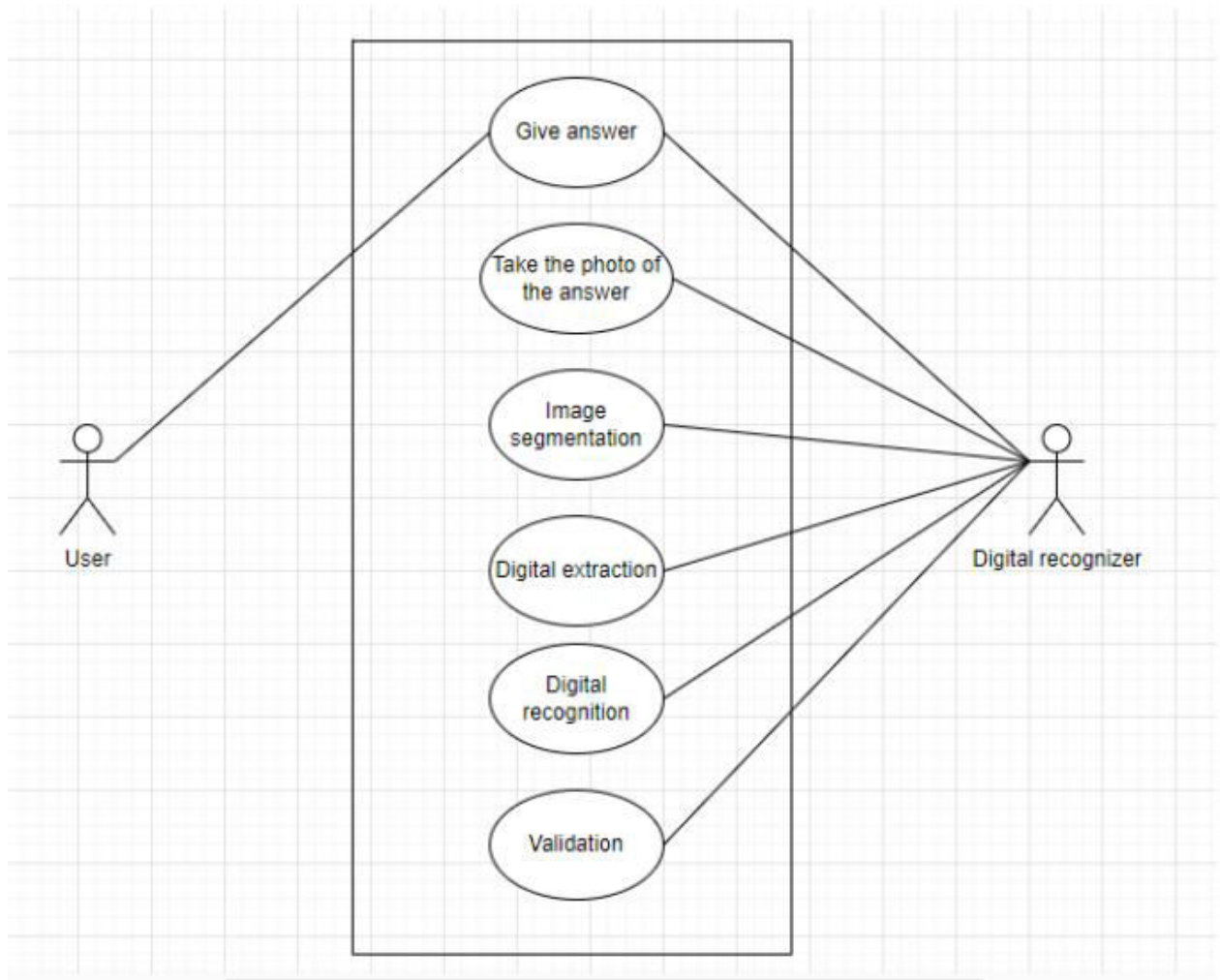


Figure 1 : Use-Case Diagram

ii. Non-Functional Requirements

Non-functional requirements are the constraints that must also be adhered to during the development. They limit what resources can be used and set bounds on aspects of the software's quality. The non-functional requirements are divided into several groups: The first group of categories reflects the four of the most important qualities attributes.

Non-functional requirements define the overall qualities or attributes of a system. So, the non-functional requirements for our system are as follows:

- Usability: The users should easily be able to write the expression with a pointing

device.

- Performance: The system should perform in terms of valid input from the user and give the accurate value of the expression.
- Reliability: The system should provide a high degree of reliability.
- Security: The system should also be secure.

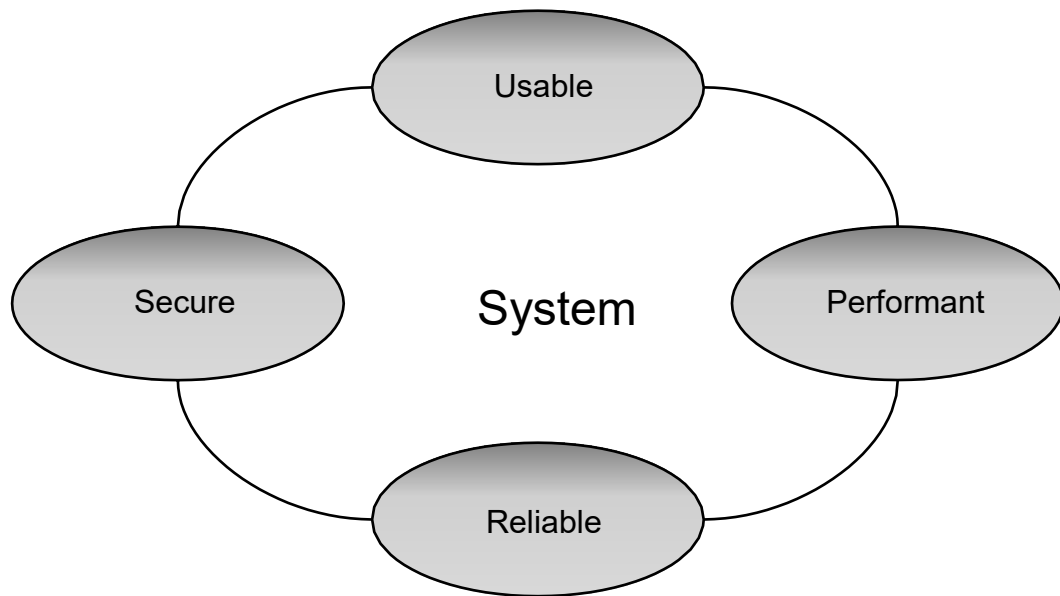


Figure 2: Non-Functional Requirements

3.1.2 Feasibility Study

A feasibility study is an examination that considers all of the pertinent project-related aspects, such as economic, operational, technical, and scheduling issues, to determine the possibility of the project's successful completion.

i. Technical feasibility

Due to the project's straightforward interface, it is technically viable. Tools that are appropriate and convenient for this project's development are being employed. This group's members are all well-known in their respective fields.

ii. Operational feasibility

Assuming that users of this software have a solid understanding of computers. The majority of users can utilize this program.

iii. Legal and feasibility

This system doesn't violate any laws. The code that is used isn't copied from any resource. The data provided to this system are from free sources. This proposed project doesn't conflict with any legal requirements like zoning laws, data protection laws or social media laws.

iv. Economic feasibility

Developed system is economically feasible. It can be developed in a simple PC or laptop which is easily available. System is built in open source language so; development is free of cost. All the references and resources used are freely available on the internet. So, the developed system is economically feasible.

v. Schedule feasibility

This project is practical in terms of scheduling considering that we have enough time period and the required schedule which helps to carry out this project more efficiently and effectively.

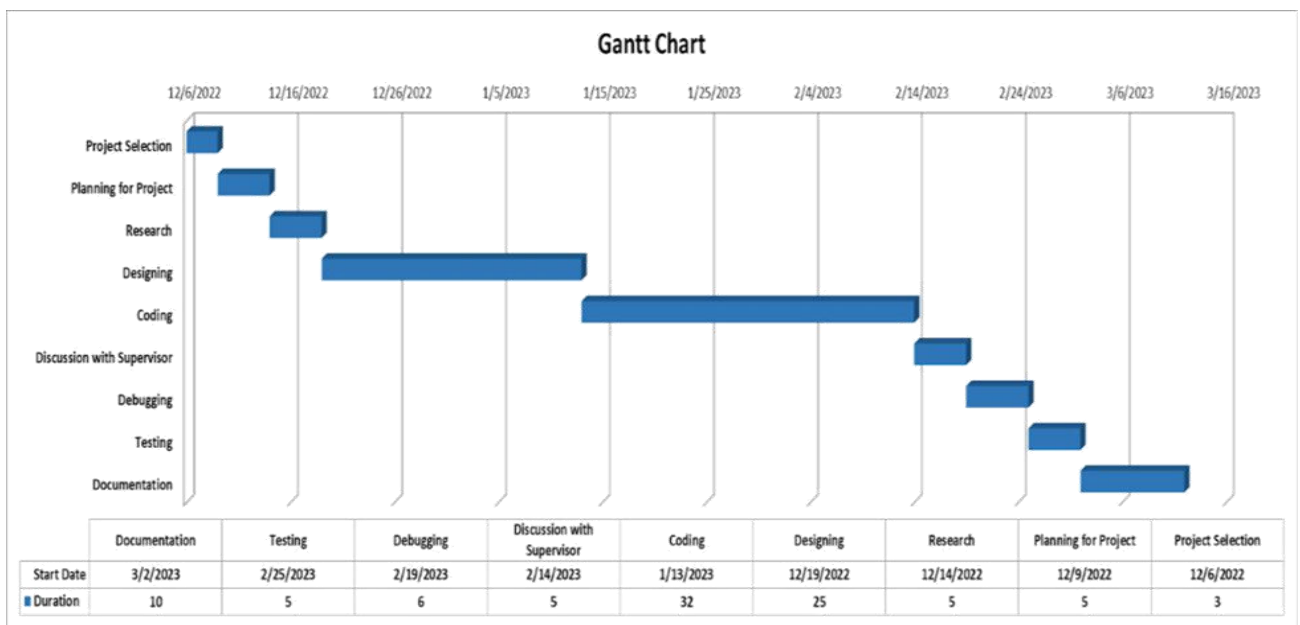


Figure 3: Gantt chart

3.1.3 System Requirements:

This section will provide the user the required specification of the hardware and the software components on which the proposed system is to be implemented.

i. Hardware Requirements:

Compatibility: Compatible with all Window PCs

Hardware specification: 512+ RAM Minimum, 100 MB free space in the hard disk, and graphics cards(optional), normal pointing device.

ii. Software Requirements:

This subsection will provide the versions of software applications that must be installed. The software requirements are as follows :-

Operating system: Windows 7, 8/10

Programming language: Python 3+

Libraries: OpenCV, keras, TensorFlow, Tkinter

CHAPTER 4

System Architecture

4.1 Design

4.1.1 System Architecture

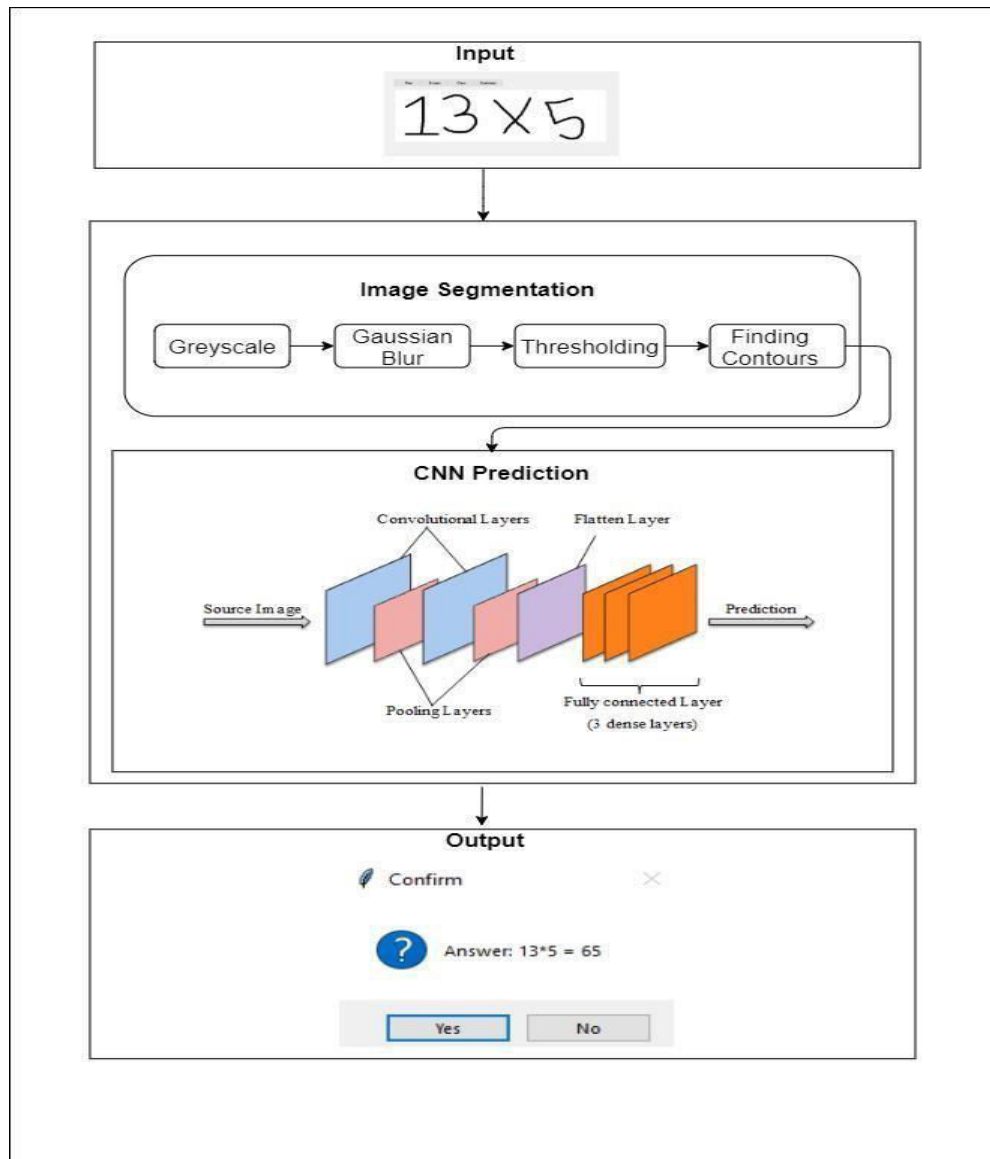


Figure 4: System architecture

4.1.2 FlowChart Diagram

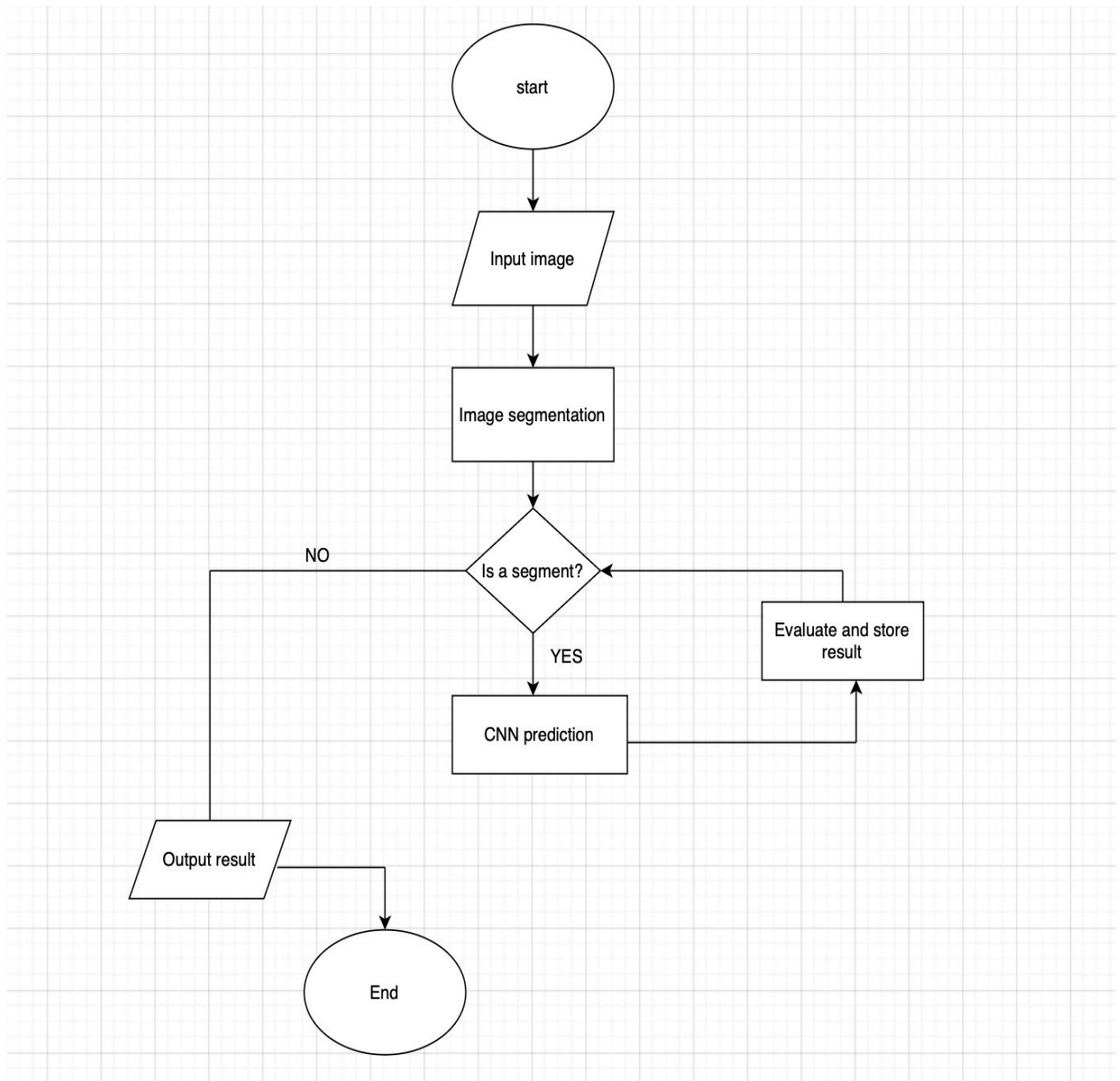


Figure 5: System Flowchart

4.1.3 Sequence Diagram

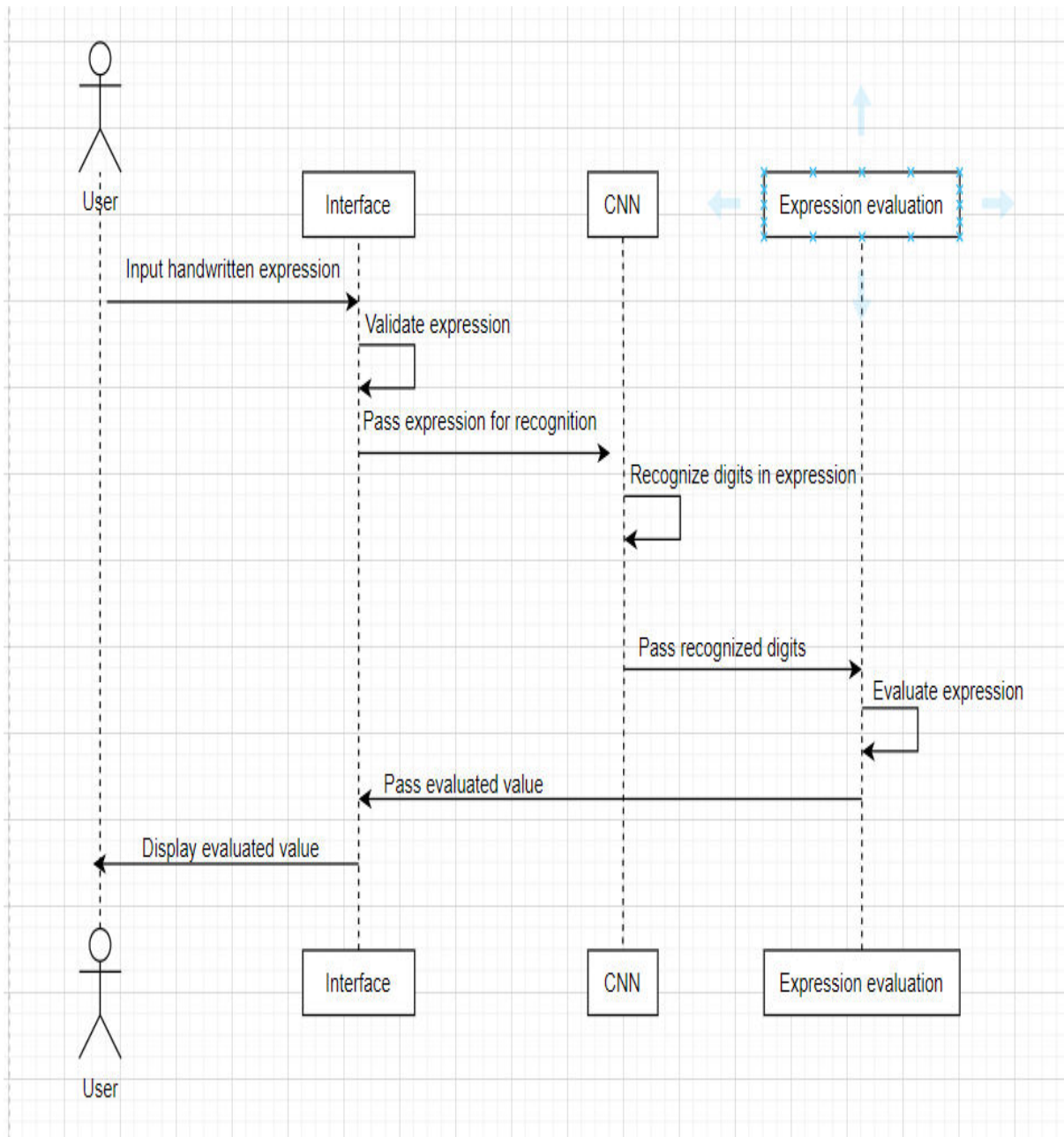


Figure 6: Sequence Diagram

4.1.4 Activity Diagram

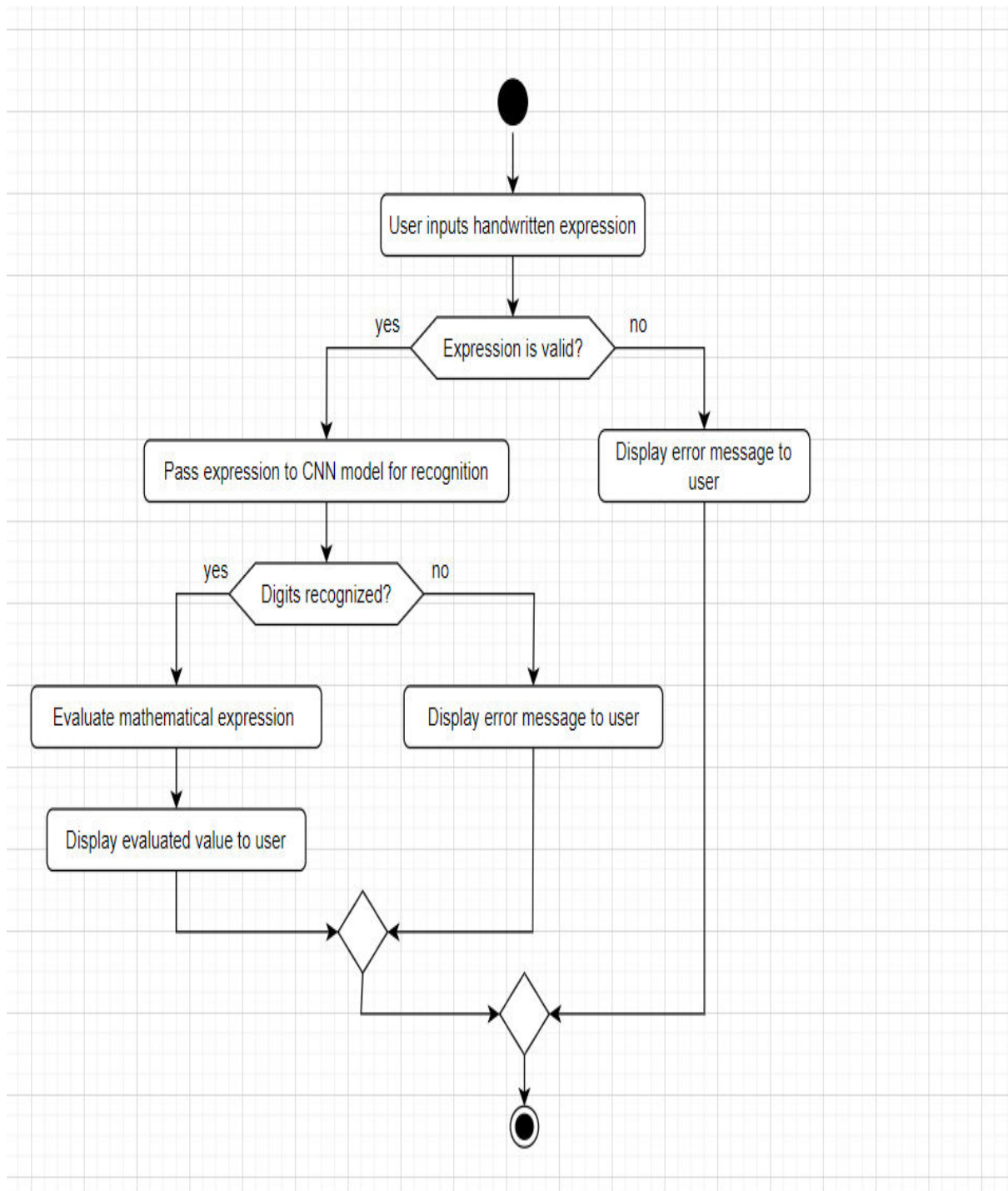
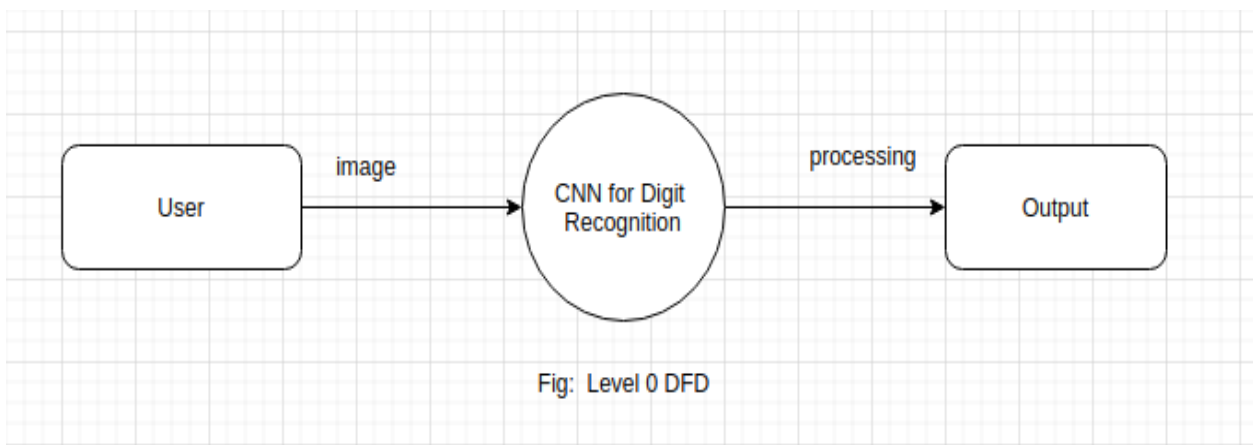


Figure 7: Activity Diagram

4.1.5 DFD Level

DFD LEVEL 0

It is also known as a context diagram. It provides an abstract view, showing the system as a single

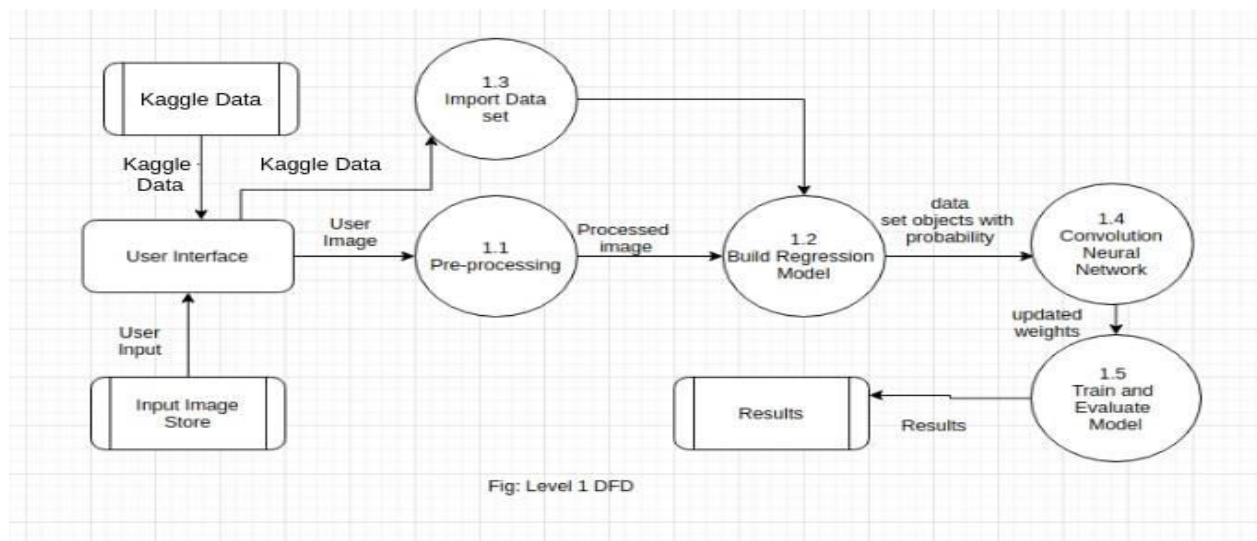


process with its relationship to external entities.

Figure 8: DFD LEVEL-0

DFD LEVEL 1

Here, the context diagram is decomposed into multiple processes. In this level, the function of the system is highlighted and the high-level processes of 0-level DFD are broken down into sub-



processes.

Figure 9: DFD LEVEL-1

4.2 Algorithm Details

4.2.1 Convolutional Neural Network:

Convolutional neural networks (CNNs) are a type of deep neural network that use a mathematical operation called convolution in at least one of their layers. This makes them different from traditional neural networks, which use general matrix multiplication. CNNs are commonly used for visual imagery analysis, and they have applications in various fields, including image and video recognition, recommender systems, medical image analysis, natural language processing, and financial time series.

One of the advantages of CNNs is their ability to use hierarchical patterns in data to assemble more complex patterns from smaller and simpler ones. This approach to regularization is different from that of traditional fully connected networks, which are prone to overfitting data. CNNs are also inspired by biological processes, specifically the organization of the animal visual cortex. The connectivity pattern between neurons in CNNs resembles the receptive fields of individual cortical neurons in the visual cortex, which respond only to stimuli in a restricted region of the visual field. Another advantage of CNNs is that they require relatively little pre-processing compared to other image classification algorithms. In traditional algorithms, filters were hand-engineered, but CNNs learn these filters on their own. This independence from prior knowledge and human effort in feature design is a major advantage. Overall, CNNs are a powerful tool for analyzing visual imagery and have wide-ranging applications in various fields.

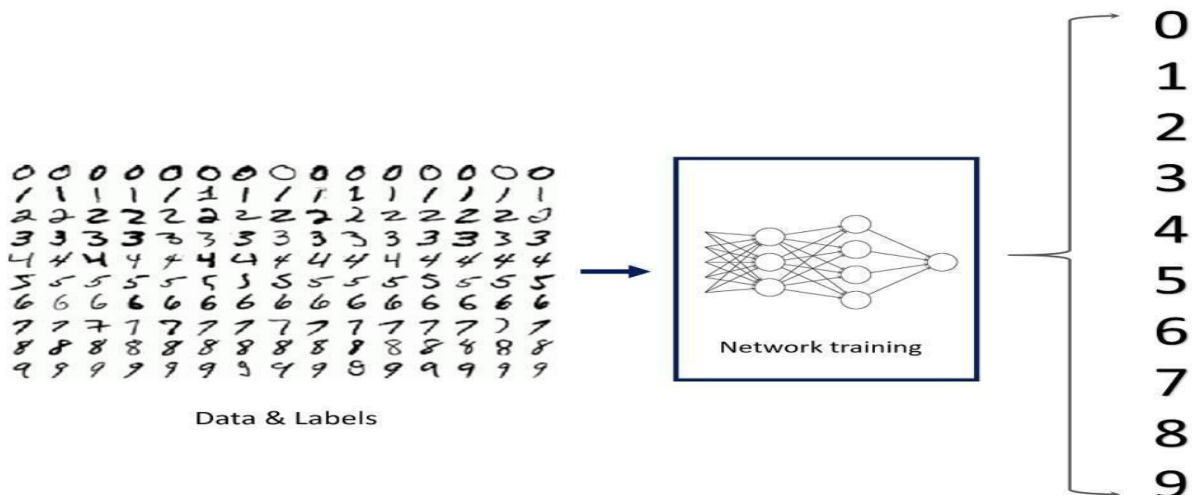


Figure 10: Convolutional Neural Network Layer

4.2.2 Convolutional Neural Network (CNN) Model:

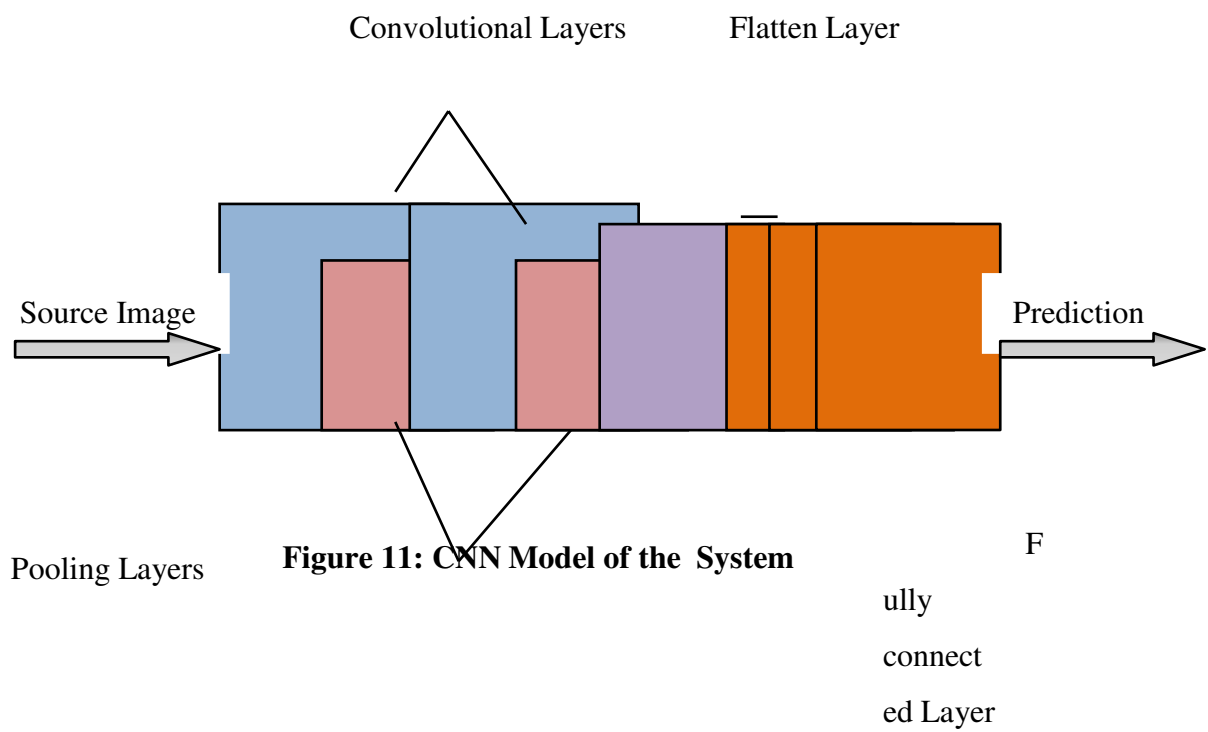
A convolutional layer, a pooling layer, and finally a fully connected layer make up a conventional convolutional neural network. Fully connected layers are then utilized to provide a prediction after the convolution layers collect features and the pooling layer reduces the image's dimension. Dropouts from the network result from neglecting part of the neurons.[3]

The first layer consists of the Convolutional layer with 30 channels, a kernel of size 5x5, and a rectifier linear unit (relu) as an activation function. This layer then feeds 30 images into the second layer. This layer is accompanied by a max-pooling layer of size 2x2 to downsample, reduce parameters, computation involved in the network and create its feature map.[3]

The next layer consists of another Convolution layer but with 15 channels, the kernel of size 3x3, and relu as an activation function. This layer is again accompanied by max-pooling of the same size as before.[3]

Dropout of 0.2 is used when feeding the feature map to the flatten layer which then transforms the two-dimensional feature map into a format suitable for a fully connected network to work on.

Finally, the fully connected layer consists of 3 dense layers each. The first dense layer consists of 128 neurons with relu activation function, the second dense layer consists of 50 neurons with relu activation function, and the last layer consists of 13 neurons with softmax as the activation function. These dense layers work together in tandem to classify the source image to any of the 13 characters (0, 1, ..., 9, +, -, *).



CHAPTER 5

IMPLEMENTATION AND TESTING

5.1. Dataset

The convolutional neural network training dataset was downloaded from the Kaggle website and made available by Kaggle user 'Xai Nano'. All of the English alphanumeric characters, arithmetic operators, set operators, predefined math functions, and other math symbols are all included in the database. The symbols are '.jpg' binary images with a black foreground and a white background. Each.jpg image is 45×45 pixels wide and tall (2025 pixels).

The original CHROME dataset's photos were processed, extracted, and changed.

The handwritten digit math system uses a portion of a dataset (0, 1,... 9, +, -, *), so a total of 47,504 pictures are used to train CNN. Each of these images is resized to 28x28 so it can be trained quickly and properly on the neural network.

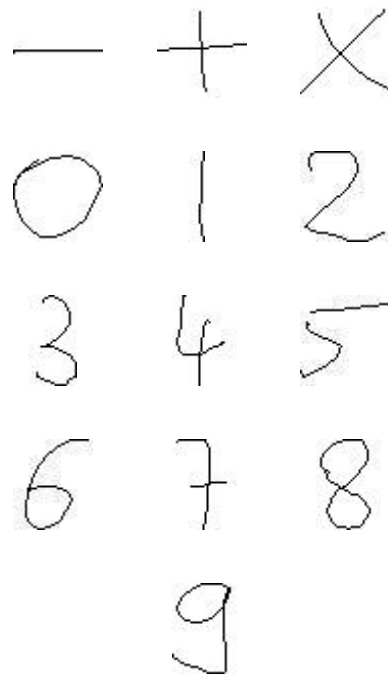


Figure 12: Dataset sample

5.2 Image Processing and Extraction of subject element

5.2.1 Grayscale

Any image must first be transformed into a grayscale image in order to be processed. This preprocessing makes it simple to process the image further with fewer calculations and no essential information loss. Specifically, "a grayscale or greyscale image is one in which the value of each pixel is a single sample representing only an amount of light, that is, it carries only intensity information."

Red, Green, and Blue make up an image's three channels ordinarily, therefore the gray scaling technique basically reduces a three-channel image to a single channel that only contains shades of gray, with Black being the darkest and White being the lightest.

Conversion of the RGB image into the grayscale image can be done using the Average Method. According to the Average method, the corresponding grayscale value can be obtained by calculating the average of RGB intensities of a pixel. This applies to every pixel in the image and finally, the whole image is converted into grayscale. However, the python cv2 module has `cv2.cvtColor()` function to convert any image into grayscale image.[8]



Figure 13: Normal image



Figure 14: Grayscale image

5.2.2 Thresholding

Thresholding is a technique for separating the foreground from the background in grayscale photographs. When a grayscale image is run through the Thresholding function, which accepts a threshold value and a max value (usually 255), a binary image (just two colors - mostly black and white) is created. Every pixel in the image has its grayscale value compared to a threshold value; if the threshold value is greater, the pixel is then given the maximum value, and if it is less than or equal to the threshold value, the pixel is given the value of 0. As a result, the final image is made up of pixels with values of max or 0, making it a binary image.

Otsu Thresholding is another much more advanced thresholding method. This method thresholds the grayscale image by going through every threshold value and finding the one which has the minimum within-class variance. [5]

Finding within-class variance

Choose a certain threshold value and segment the image into two groups (foreground and background). For both, the groups find weight (W), mean (μ), and variance (σ^2); finally, the within-class variance is given by:

$$\sigma_w^2 = W_f \sigma_f^2 + W_b \sigma_b^2$$

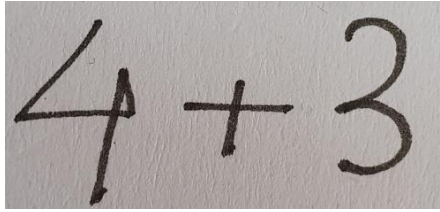


Figure 15: Original image

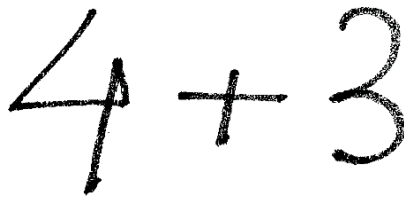


Figure 16: Binary Thresholding



Figure 17: Binary inverse Thresholding

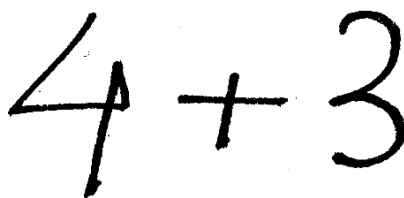


Figure 18: Otsu thresholding

5.2.3 Gaussian Blur

When used correctly, Gaussian Blur eliminates visual noise and smooths out the image. When removing primary topics from photographs with a lot of visual noise, it is a crucial step. It also goes by the name of a low pass filter since it cuts down high-frequency components. For applying Gaussian Blur to a picture, use the `cv2.GaussianBlur()` method from the `cv2` module. [4]

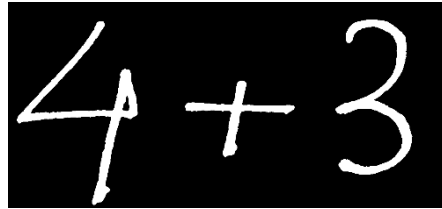


Figure 19: Otsu plus Binary Inverse Thresholding after applying Gaussian Blur (15x15 kernel)

5.2.4 Contours

Contours are the curves containing all the points along the boundary with uniform (same) intensity in the image. Contours can be found much easier when the source image is binary which can be obtained by applying any threshold methods. The cv2 python module has cv2.findContours() function to find contours in the binary image. [6]

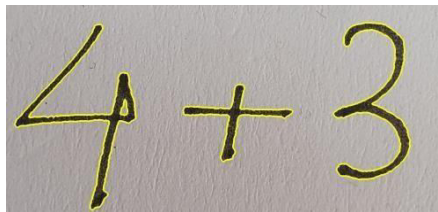


Figure 20: Individual Contours

5.2.5 Modeling a Convolutional Neural Network to classify handwritten digits

To classify the handwritten digits, a 5-layer Neural network is used consisting of one input layer and one output layer. It is the input layer of the model. Three layers are hidden.

It uses two activation functions:

5.2.5.1 Rectifier Linear Unit (ReLU)

5.2.5.2 Softmax

5.2.6 Rectifier Linear Unit (ReLU)

In the context of artificial neural networks, the rectifier is an activation function defined as the positive part of its argument:

$$f(x) = f^+ = (\mathbf{0}, f),$$

Where x is the input to a neuron. Rectified linear units find applications in computer vision and speech recognition using deep neural nets.

5.2.7 Soft plus

A smooth approximation to the rectifier is the analytic function:

$$f(x) = \frac{1}{2} \left(x + \sqrt{x^2 + 4} \right)$$

which is called the soft plus or SmoothReLU.

5.2.8 Softmax

The softmax function in mathematics normalizes a vector of K real numbers into a probability distribution made up of K probabilities proportional to the exponentials of the input numbers. It is sometimes referred to as the softargmax [3] or normalized exponential function.

Softmax is often used in neural networks, to map the non-normalized output of a network to a probability distribution over predicted output classes.

The standard (unit) softmax function is defined by the formula:

$$\sigma(\mathbf{z})_i = \frac{e^{z_i}}{\sum_{j=1}^K e^{z_j}} \text{ for } i = 1, \dots, K \text{ and } \mathbf{z} = (z_1, \dots, z_K) \in \mathbb{R}^K$$

5.3 Tools used

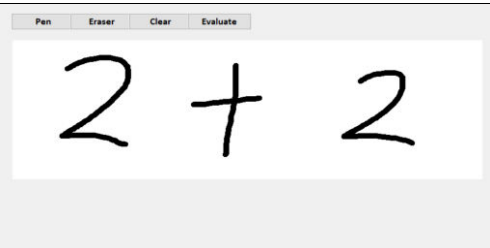
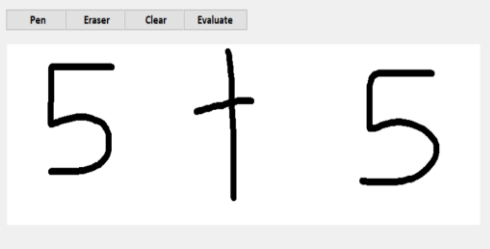
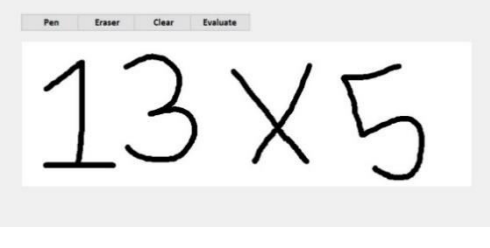
- Python: Implement the Convolutional Neural Network.
- OpenCv: Used for image segmentation (processing).

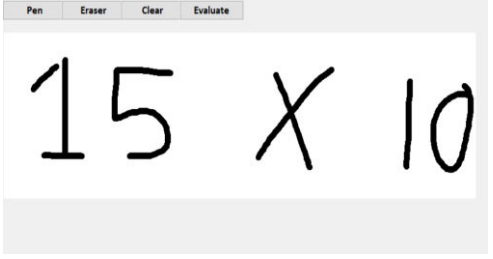
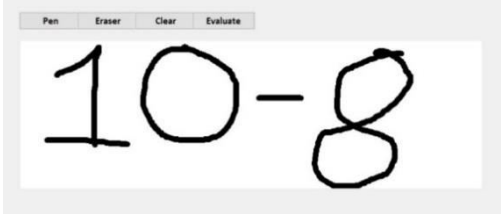
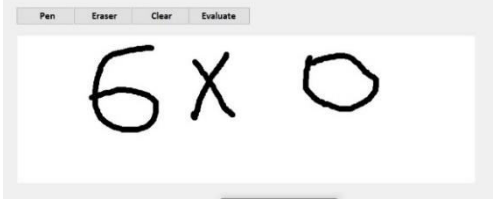
- TensorFlow: Used with Keras fronted for creating our own neural network model for predictions.

5.4 Testing

During the learning phase, the model was thoroughly tested to enhance prediction accuracy. To ensure superior image quality, the final program was evaluated on similar images captured with a high-quality mobile phone, and it accurately predicted each handwritten digit (0 to 9) and basic mathematical symbols ('+', '-', and 'x'). In later stages of development, the model underwent numerous tests against digits and symbols drawn with a mouse, and the evaluations remained consistently positive with accurate results at each stage.

Table 1: Test Cases

S.No	Input	Expected Outcome	Actual Output	Test Result
1		4	4	Pass
2		10	10	Pass
3		65	65	Pass

4		150	150	Pass
5		2	2	Pass
6		0	5	Fail

CHAPTER 6

6.1 Conclusion

On the basis of this, we draw the conclusion that modern image processing and neural network technology can both completely replace calculators for simple calculations. There is a lot of potential here because these technologies can be used to solve complex mathematical equations, which are sometimes difficult and confusing to enter on a calculator.

There are drawbacks to this system, though. These days, everyone has a cell phone, which puts them at risk of being overly dependent on the system. Even though the neural network was trained on tens of thousands of data samples, the system is not perfect and occasionally produces incorrect answers. The system is still susceptible to errors, thus the average user should be constantly aware of this.

Technically speaking, image processing and convolutional neural networks are used for more than only recognition tasks. While Convolutional Neural Networks employ the kernel to extract features, alternative algorithms, such as HOG (Histogram of Oriented Gradients), which use the intensity gradient distribution to extract features from the picture, are also very effective for character recognition applications.

Since always, it will be difficult to eliminate outside noise, segment changeable images, and recognize partial characters. However, since neural networks and image processing advance every day, answers to these problems are not that far away.

6.2 Future Enhancement

At present, the system is capable of recognizing only basic arithmetic symbols such as plus, minus, and multiplication. However, in the future, we aim to expand the system to recognize more complex mathematical notations, including scientific symbols like square roots, fractions, and other advanced operators, which are often used in scientific and technical applications.

REFERENCES

- [1] ["Remark Office OMR, by Gravic \(Principia Products\), works with popular image scanners to scan surveys, tests and other plain paper forms".](#)
Omr solutions.com.
- [2] Schantz, Herbert F. (1982). The history of OCR, optical character recognition. [Manchester Center, Vt.]: Recognition Technologies Users Association.
- [3] Valueva, M.V.; Nagornov, N.N.; Lyakhov, P.A.; Valuev, G.V.; Chervyakov, N.I. (2020). "Application of the residue number system to reduce hardware costs of the convolutional neural network implementation". Mathematics and Computers in Simulation.
- [4] 'Smoothing Images' (2020). Gaussian Blurring. OpenCV docs. Available at: https://docs.opencv.org/master/d4/d13/tutorial_py_filtering.html .
- [5] Dr. Greensted, Andrew. 2010. Otsu Thresholding. Available at: <http://www.labbookpages.co.uk/software/imgProc/otsuThreshold.html>
- [6] 'Contours: Getting Started' (2020). OpenCV docs. Available at: https://docs.opencv.org/3.4/d4/d73/tutorial_py_contours_begin.html
- [7] Nano, Xai. 'Handwritten math symbols dataset' (2017). Kaggle. Available at: <https://www.kaggle.com/xainano/handwrittenmathsymbols>
- [8] Lech, Thomas. 'Image processing for text recognition' (2017). MachineLearning-Blog. Available at: <https://blog.mathocr.com/2017/06/25/image-processing-for-text-recognition.html>

Appendix

Snapshots

