Project and Professionalism

(6CS020)

Smart Calculator using OCR

Student Id : 2039244

Student Name : Sandip Thapa

Supervisor : Mr. Manish Deuja

Cohort/Batch : 4

Words Count :

Declaration

Abstract

Contents

1. Introduction 1
   1.1. Introduction to OCR 1
   1.2. Problem Domain 1
   1.3. This project as solution 2
2. AI aspect of the project 3
   2.1. Computer Vision 3
   2.2. History of computer vision 3
   2.3. How computer vision works 3
   2.4. Applications of Computer Vision 4
   2.5. Basis of Machine learning 4
      2.5.1. Supervised Learning 4
      2.5.2. Unsupervised Learning 4
      2.5.3. Reinforcement Learning 5
   2.6. Math behind Computer vision 5
3. Algorithm used in the project 6
   3.1. Convolutional Neural Network 6
      3.1.1. Input Layer 7
      3.1.2. Convolution Layer 7
      3.1.3. Pooling Layer 9
      3.1.4. Fully Connected Layer 10
4. Literature Review 11
   4.1. Development of OCR 11
      4.1.1. First Generation OCR System 11
      4.1.2. Second Generation OCR System 11

4.1.3. Third Generation OCR System 11
4.1.4. Fourth Generation OCR System 11
4.2. Working Principles 12
4.3. Handwritten Character Recognition 13
4.4. Similar Systems 14
4.4.1. Photo Math 14
4.4.2. Math Solver 14
4.4.3. Mathway 15
4.5. Review of Similar Systems 15
4.6. Why Tesseract OCR? 16
5. Artefact 17
5.1. FDD 17
5.2. SRS Legend 17
5.3. SRS Table 18
5.4. System Modeling 19
5.4.1. Context Modeling 19
5.4.2. Structural Modeling 21
5.4.3. Process Modeling 22
5.4.4. UI Model 24
6. Full details of Artefact 25
6.1. Development Methodology 25
6.2. Preferred Methodology 26
6.3. Techniques and Tools 27
6.4. Testing 28
6.5. Probable Issues During the project 29

6.6. Plan/Schedule 30
7. Model Development 1
7.1. Data Collection 1
7.2. Data Training 1
7.3. Algorithm Comparison 1
7.4. Performance Analysis 1
7.5. Confusion Matrix 1
7.6. ROC Curve 1
References 2

Table of Figures:

Figure 1: Architecture of CNN 6
Figure 2: Convolution Operation 8
Figure 3: Max Pooling vs Avg Pooling 10
Figure 4: Architecture of Tesseract OCR (Patel & Patel, 2012) 12
Figure 5: Segmented Character from input image 13
Figure 6: Downsampled Character 13
Figure 7: PhotoMath Application 14
Figure 8: Math Solver Application 14
Figure 9: Mathway Application 15
Figure 10: Use Case Diagram for Basic Calculation System 19
Figure 11: Use Case Diagram for Image Capturing System 20
Figure 12: Use Case Diagram for Character Recognition System 20
Figure 13: Class Diagram 21
Figure 14: Context Diagram for Image Capturing System 22
Figure 15: Context Diagram for Character Recognition System 23
Figure 16: History Screen 24
Figure 17: Calculator Screen 24
Figure 18: Solution Screen 24
Figure 19: Navigation Screen 24
Figure 20: Main Screen 24
Figure 21: Incremental Model Visualization (EduCba, 2020) 26
Figure 22: Gantt-Chart 30

1. Introduction

1.1. Introduction to OCR

Optical Character Recognition is a technology that allows machine to recognize the text whether it is scanned or printed text images or handwritten text. The machine can do further processing on the data extracted from that text. It can be considered same as the combination of human eye and mind. An eye can see the text from some source but mind is the one that actually processes and interprets that text.

OCR system is made up of combination of both hardware and software. Hardware such as, optical scanner or some specialized circuit board is used to read or extract text. And software does the advance processing.

The most common use of OCR is to convert hard copy documents into softcopy files such as PDFs. This will make easier to edit the document. There are many other applications of OCR, such as: image text extraction, extracting texts from scanned documents, License plate recognition and answer paper checker. (Patel, et al., 2012)

The main concept of this project is to implement OCR in an android based calculator application. The objective of this application is to scan handwritten or printed numbers and perform mathematical calculations on it.

1.2. Problem Domain

In this modern era of technology, people don't want to waste time in less important tasks like performing mathematical calculation in traditional ways which are generally time consuming. For that they have calculators, but they still need manual input from the users. They need to follow some sets of guidelines and go through each processes to get correct answer. It would be much more time saving if user don't have to manually enter the data for calculation. If the calculator gets the data with just one click, it will definitely be more efficient and time saving.

Besides, while checking the answer paper, teachers need to solve the question first by themselves and then only can correct the paper. Students are also in dilemma after solving a problem whether their answer was correct or not. It would be great help for them if they knew if their answer was correct.

1.3. This project as solution

The main objective of this project is to develop an app that can perform mathematical calculations without even having to type to give input. It scans for a mathematical problem mentioned in any paper and gives back the answer. Since it is a mobile application, user can use their camera to scan the problem. The app then extracts the problem from paper using OCR and perform operations on it and return the result. Teachers and students can check whether their answers were right or not just by a click. Since this app knows all the mathematical rules and principles, users don't need to bother about remembering rules like BODMAS, which they had to keep in mind if they were to perform calculation manually in traditional calculators.

2. AI aspect of the project

2.1. Computer Vision

Computer vision is one of the field of artificial intelligence that works on how computers can gain high-level knowledge from digital images or videos. In other words, computer vision is a field that enables computers to see, identify and work on images in the same way as we human do and even surpass human ability in many cases (Huang, 1996). In computer vision, we train machines to interpret and understand the visual world by providing computers human-like perception capabilities (Sebe, et al., 2005). For that, we use deep learning models on digital images and videos due to which machines can correctly identify and classify objects.

2.2. History of computer vision

Studies in computer vision started only in the 1950s. They made use of some of the first neural networks to detect the edges of an object and to sort simple objects into categories. It was only in the 1970s, the first commercial use of computer vision was done. The system interpreted digital or handwritten text using Optical Character Recognition. As the internet started to mature from the 1990s, large set of images were available online for analysis. These growing datasets helped to train the models and make it possible for machines to identify specific objects from photos and videos.

Today, mobile technology with built-in camera, cheap and powerful computers and new algorithms like convolutional neural network have converged to bring revolution in computer vision. The effect of these advancements have increased the accuracy rates for object detection and classification using computer vision have gone from 50 percent to 99 percent in less than a decade. (Ranjay, 2017)

2.3. How computer vision works

The very first step in computer vision is feature engineering. As part of this, the computer acquires images and videos and convert them into an array of pixels. Then the model identifies the features such as edge and corner of an object. Before predicting the objects it is necessary that we train the model by feeding thousands of such labeled data. Once after the model analyzes pixels, computer vision uses neural network to predict the content of the image.

2.4. Applications of Computer Vision

- Health Care

  In health care sector, computer vision enabled machines can help doctors to detect different symptoms of diseases in MRI scans and X-rays.

- Facial Recognition

  Facial Recognition is one of the emerging application of computer vision. From general social networking sites to law enforcement agencies, facial recognition technology is being used to identify the faces. Moreover facial recognition lock system in gadgets are being popular these days.

- Autonomous vehicles

  In autonomous vehicles, the cameras attached capture the surroundings. Then the computer vision software process it to find the traffic signs, roads and other vehicles. Based on these observations the vehicle identifies its path and avoid the obstacles.

2.5. Basis of Machine learning

Computer vision is based on machine learning technique of Artificial Intelligence. There are mainly three types of machine learning. Namely, supervised learning, unsupervised learning and reinforcement learning.

2.5.1. Supervised Learning

Supervised learning is the machine learning technique in which we train the machine using well labeled data. This means data is already provided with correct answers. This learning technique allows us to collect data and predict an output based on previous experiences.

2.5.2. Unsupervised Learning

Unsupervised learning is a machine learning technique in which we do not need to train the model ourselves. The model itself works to discover patterns and information on unlabeled data. Clustering is an example of unsupervised data.

2.5.3. Reinforcement Learning

Reinforcement learning is a machine learning technique about taking a suitable action to maximize the reward in a specific situation. In reinforcement learning there is no answer tagged to the training data set like in the supervised learning. Reinforcement agent itself decides what action to carry out in order to perform the given task. And in the absence of a proper training dataset, the machine is has to learn from the past experiences.

As computer vision is based on training the machine with images and videos, it mostly falls under the category of supervised learning. But the some study has shown that the machine learning strategies in computer vision are supervised, unsupervised as well as semi-supervised. Computer vision uses various machine learning algorithms like, CNN, RNN, K-Means Clustering and LSTM. These different algorithms falls under different learning category. Thus pushing computer vision to fall under multiple category of machine learning. (Khan & Al-Habsi, 2019)

2.6. Math behind Computer vision

3. Algorithm used in the project

Computer vision is a field of deep learning that deals with images on all scales. It allows our machines to understand the content in the image. The Convolutional Neural Network is used as the main architecture or the algorithm behind computer vision which is nothing but the derivative of feed-forward neural networks.

3.1. Convolutional Neural Network

In machine learning, a convolutional neural network is nothing but a class of deep, feed-forward artificial neural networks. It is mostly used in analyzing visual imagery. CNN is a neural network with convolution operation instead of matrix multiplication in at least one of the layers. Convolutional neural networks are usually composed by a set of layers that can be grouped by their functionalities.

Our computers see images as high dimensional vectors. It would take a huge amount of parameters to characterize the network. To solve this problem convolutional neural networks reduce the number of parameters and adapt the network architecture specifically to vision tasks.

The whole working mechanism can be described by the following diagram.

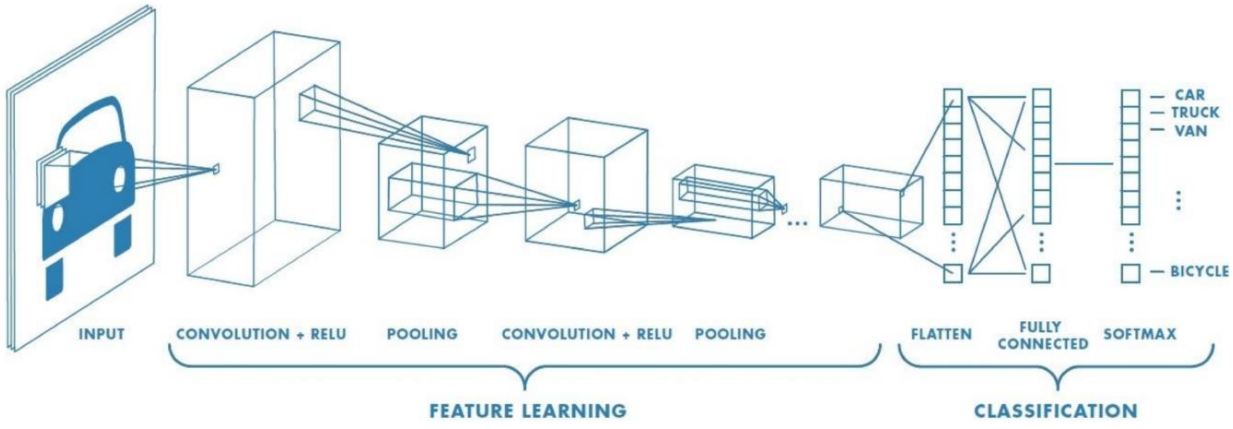


Figure 1: Architecture of CNN

3.1.1. Input Layer

The input layer holds the input of the whole CNN. In the neural network of image processing it holds the pixel matrix of the image. A CNN usually takes 3 order tensor as input. For example, an image with H rows, W columns and 3 channels(R, G, B color channels). However, higher order tensor inputs can also be handled by CNN.

3.1.2. Convolution Layer

Convolutional layer is the layer where convolution operation is carried out. The convolution layer is the core building block of CNN. The parameters consist of multiple sets of learnable filters. Every filter is small in size (width and height), but extends fully through the full depth of the input volume. During the feed-forward pass, we slide each filters across the width and height of the input volume and compute the dot products between the matrix of the filter and the input. A set of filters in every convolution layer outputs a separate 2-D activation map. After defining the stride and padding we define the convolution product between tensor and a filter. Convolution product is nothing but a sum of element wise product of 2D matrix formed out of pixels and filter.

An image, in general can be mathematically represented as a tensor with the following dimension:

$$dim(image)=(n_H, n_W, n_C)$$

Where:

n_H *: the size of the height*

n_W : the size of the width

n_C *: the number of channels*

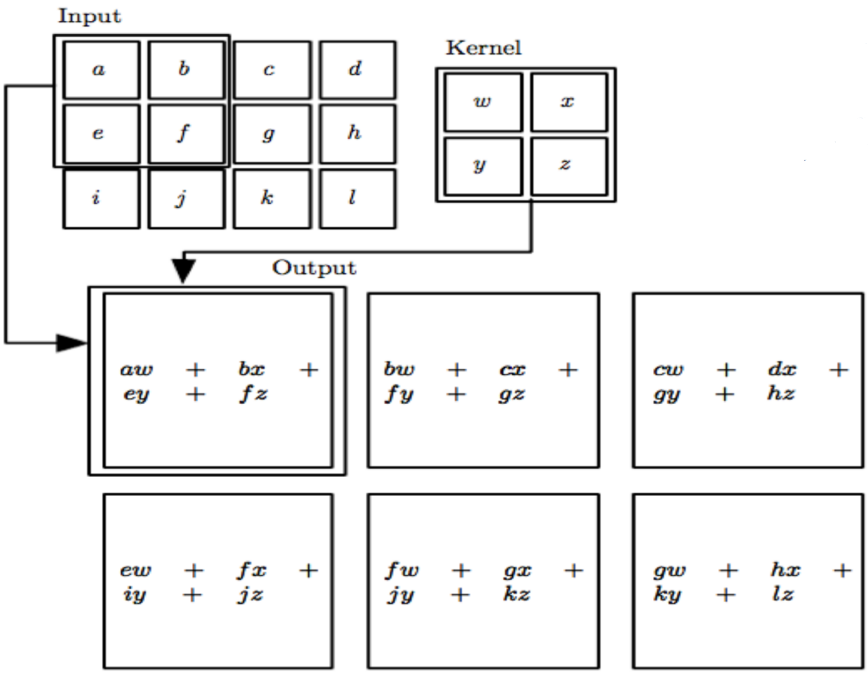


Figure 2: Convolution Operation

Mathematically summarizing the convolution layer:

- It accepts the input of size W_1 x H_1 x D_1
- Requires four hyper-parameters:
  - Number of filters, **K**
  - Their spatial extent, **F**
  - The Stride, **S**
  - The amount of padding, **P**
- The output of size W_2 x H_2 x D_2 where:
  - $W_2 = (W_1 – F + 2P) / S + 1$
  - $H_2 = (H_1 – F + 2P) / S + 1$
  - $D_2 = K$

3.1.3. Pooling Layer

The use of a pooling layer is to reduce the dimension of the input image. By sampling the convolved feature maps, the useful information of the image is preserved and the redundant data is removed, thus effectively preventing the over fitting problem and also increasing the computation speed.

In this step image is down sampled by summing up the information. The pooling action is carried out through each and every channels and thus it only affects the dimensions and keeps the channels intact.

Given an image, we slide filter through height and width of input volume with no parameters to learn. Then following a certain stride, we apply a function on the selected elements. It can be represented mathematically as:

$$\begin{aligned} dim(pooling(image)) &= (\left\lfloor \frac{n_H+2p-f}{s} + 1 \right\rfloor, \left\lfloor \frac{n_W+2p-f}{s} + 1 \right\rfloor, \mathbf{n_C}); s > 0 \\ &= (n_H + 2p - f, n_W + 2p - f, \mathbf{n_C}); s = 0 \end{aligned}$$

We generally use max pooling and average pooling in this layer. Below is the illustration of max pooling and average pooling is shown

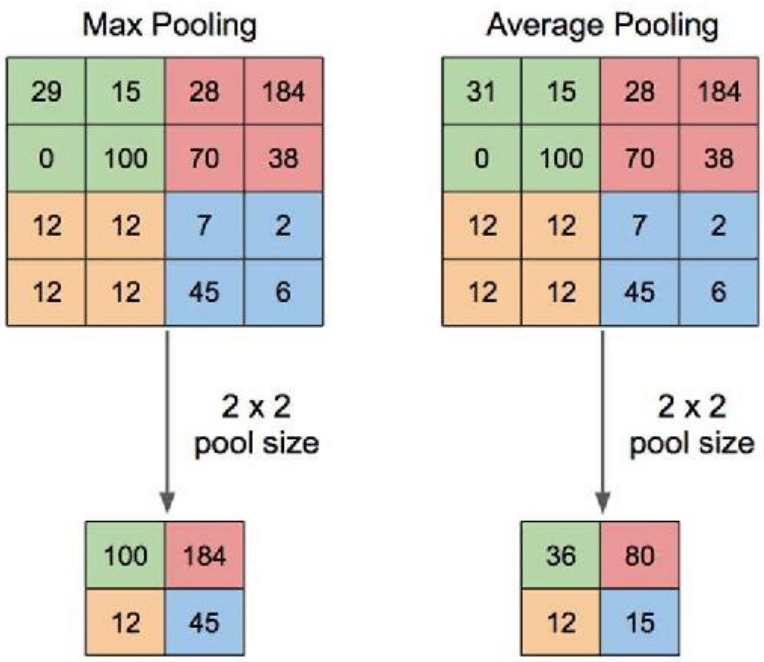


Figure 3: Max Pooling vs Avg Pooling

3.1.4. Fully Connected Layer

Fully connected layer is usually the last layer of the network. This layer combines the information gathered from the former layers to achieve the explicit expression of classification. The input of the fully connected layer comes in the form of output from the final pooling layer. That output from pooling layer is first flattened and then fed into the fully connected layer in the form of 1D array. (Wu, 2017)

4. Literature Review

4.1. Development of OCR

4.1.1. First Generation OCR System

Character Recognition was first originated in early 1870s with the invention of retina scanner. The first generation OCR appeared only in the beginning of 1960s with the development of the digital computers. This generation machines could read symbols specially designed for them. The first commercialized OCR of this generation was *IBM 1418*, which was designed to read special IBM font, *407*. The recognition method was template matching, which compares the character image with a library of prototype images for each character of each font . (Shodh Ganga, 2015)

4.1.2. Second Generation OCR System

“This generation machines were able to recognize machine printed as well as hand written characters. But the character set was limited to numerals and a few letters and symbols. Such machines appeared in between middle of 1960s to early 1970s.” (Shodh Ganga, 2015)

4.1.3. Third Generation OCR System

This generation OCR systems mainly focused on overcoming the challenges like poor document quality, large printed and hand written character sets. Low cost and high performance were also important concerns . (Shodh Ganga, 2015)

4.1.4. Fourth Generation OCR System

This generation system focuses on complex documents which contain texts, graphics, tables, mathematical symbols, unconstrained handwritten characters, low-quality noisy documents and many more. (Shodh Ganga, 2015)

In this project we are going to use OCR to recognize handwritten or machine printed numerals and mathematical symbols. We can then perform mathematical calculations on retrieved data. There already exists some products that use OCR to do mathematical calculations. Some of them are presented below in the section *4.4*.

4.2. Working Principles

The objective of OCR is to extract the text and convert it into editable form. For that, a document is first scanned using an optical scanner which produces an image form of the document. Now this text image is converted into editable character code such as ASCII. The basic working principle of OCR can be show as following figure.

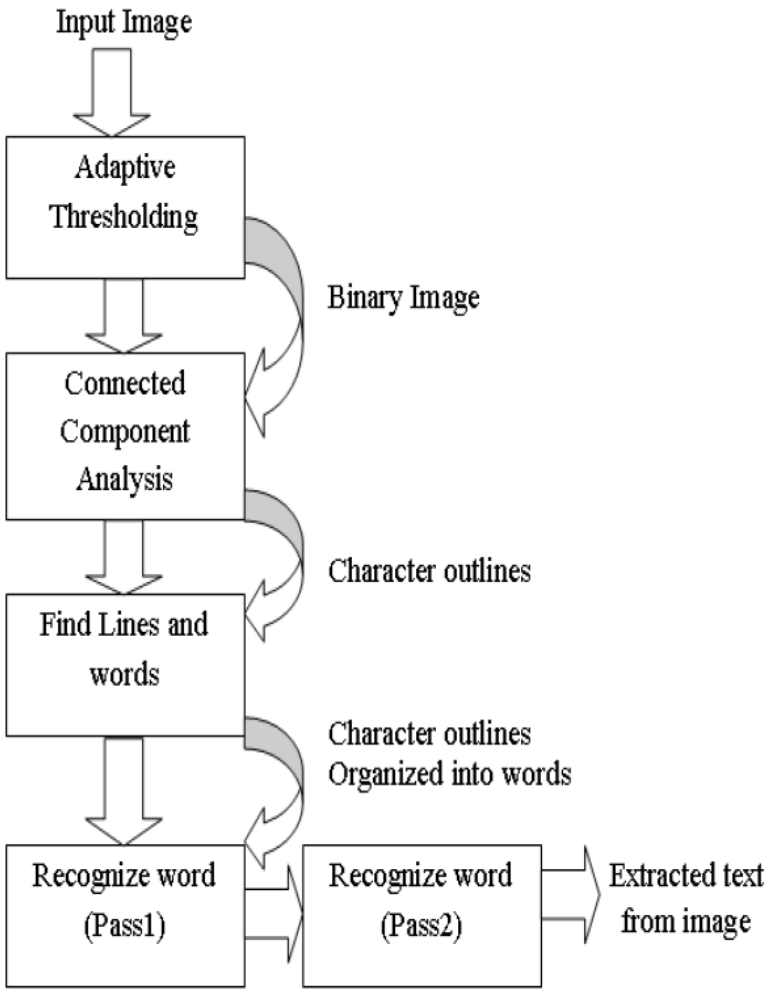


Figure 4: Architecture of Tesseract OCR (Patel & Patel, 2012)

4.3. Handwritten Character Recognition

Tesseract performs well for printed numbers with the detection rate more than 85% for the fonts within its database. But for the handwritten numbers it drops to about 50%. The main reason for this difference in result is due to the variation in the size of numbers that are handwritten and also due to the lack of matching fonts in its database.

To overcome this problem a machine learning algorithm based on Support Vector Machines (SVM) can be applied. This algorithm analyzes data and recognize patterns. This algorithm first converts character images into vector form. After using line segmentation region labels are used to determine the bounding box for each individual characters. A small amount of padding is added to the border, as shown in Figure 2. The segmented character is now downsampled to 32x32 pixels and then divided into 64 4x4 regions. The count in each region is the determined vector value, as shown in Figure 3. This conversion thus results in a 64 dimensional vector for each character image. (Sikka & Wu, 2012)

Figure 5: Segmented Character from input image

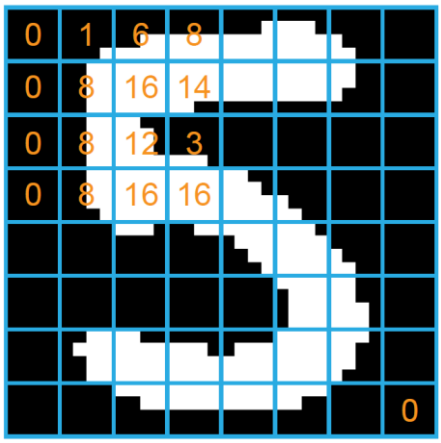


Figure 6: Downsampled Character

4.4. Similar Systems

4.4.1. Photo Math

Photomath is one of the best math problem solver application. It uses phone's camera to capture a picture of the math problems. Then the picture is scanned by the application. The application uses advanced OCR technology in order to recognize both, handwritten and printed characters. The recognized characters are then processed through Photomath's own algorithm that examines every character and determines the formula for the scanned problem. Finally, a problem solving algorithm is applied to the formula and the solution is provided with every solving steps. (Photomath, 2020)

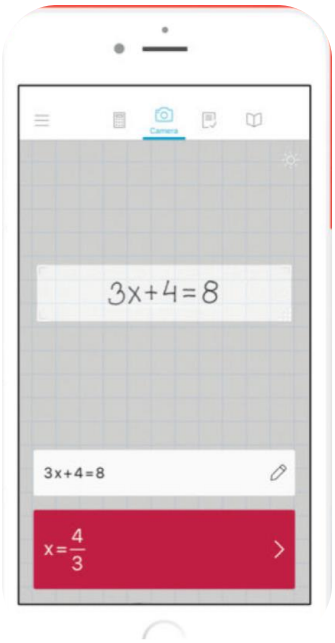


Figure 7: PhotoMath Application

4.4.2. Math Solver

Microsoft Math Solver can not only solve simple mathematical calculations, it can solve various math problems like quadratic equations, calculus and statistics. The application can also show graphs of the equations.

We can either type our problem query using a scientific calculator in the application or draw it on the phone's screen. But most importantly we can just use our phone's camera to scan the problem on our books or on the copy written by us. (Microsoft, 2019)

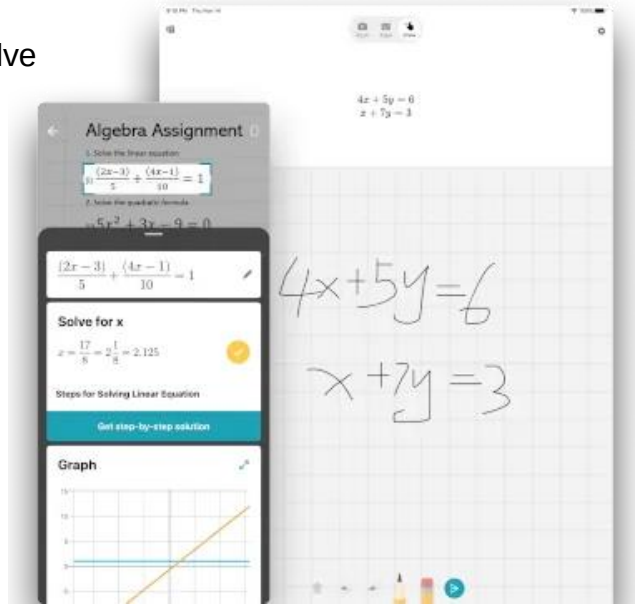


Figure 8: Math Solver Application

4.4.3. Mathway

Mathway is little bit different system than the other two. It works by letting the user to choose the field of mathematics of which the problem is to be solved and then allows user to input the problem by either typing or scanning. The problem is then processed and provides the result in conversational style like the chat bot does. (Mathway, 2020)

This system implements Lexical analysis in order to solve the problem. It first breaks the problem into tokens. In an expression 1+2, the tokens are 1, + and 2. Then the tokens are fed into the parser, which has the knowledge about relationships between tokens and can call the appropriate function which in this case is add.

Figure 9: Mathway Application

4.5. Review of Similar Systems

All of the above mentioned systems have their separate math content team. Because of this there is solution to every math problem from arithmetic to calculus. These systems read and solve mathematical problems by just using the camera of mobile phones. The most astonishing feature of these systems is that they provide step-by-step solutions too. We can even choose multiple explanation methods for same problems. Moreover, they also provide animated calculation steps.

The only bad aspect or the limitation of these systems is that they support only English language. They can only perform calculations on English numerals. This is the aspect where my project is going to work on. My smart calculator will be able to perform calculations on Nepali numerals too.

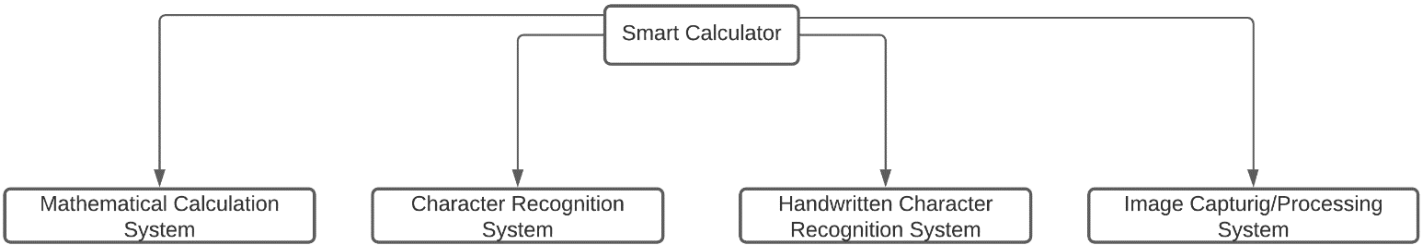
4.6. Why Tesseract OCR?

Some of the reasons to use Tesseract OCR are as follows:

- It is platform independent
- Supports multiple languages (Google Open Source, 2020)
- High accuracy
- Open source
- Ease of access and use
- Tesseract has the font accuracy in the range of 85-90%

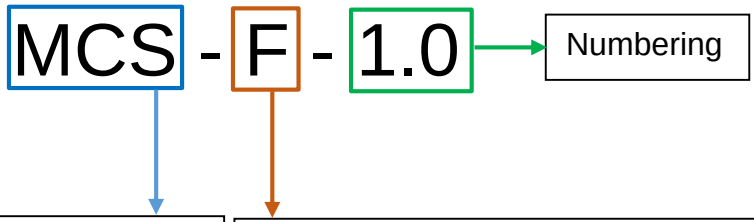
5. Artefact

5.1. FDD



5.2. SRS Legend

Legend



Sub Systems:

MCS : Mathematical Calculation System

CRS : Character Recognition System

HCS : Handwritten Character Recognition System

IPS : Image Processing System

Types of Requirements:

F : Functional Requirements

NF : Non-Functional Requirements

UR : Usability Requirements

5.3. SRS Table

Requirement Code	Requirement Description
MCSF **1.0**	The system should use the built in ALU to perform arithmetic calculations.
MCSNF 1.1	The system should provide the result in less than 1.5 seconds.
MCSNF 1.2	The system should follow BODMAS rule for calculations.
MCSUR 1.1	The system should have “All Clear” button to reset the entry.
MCSUR 1.2	The system should have “Clear Entry” button to erase the latest entry.
MCSUR 1.3	The system should display result in bigger font than the input.
MCSF **2.0**	The system should be able to perform scientific calculations.
MCSNF 2.1	The system should be able to point out the error if any occurs.
MCSNF 2.2	The scientific notations should be placed separately.
MCSF **3.0**	The system should keep records of the calculations history.
MCSNF 3.1	The system should display maximum five history records.
MCSNF 3.2	The system should have clear history option.
MCSUR 3.1	The system should have black background with white text color.
CRSF **1.0**	The system should recognize numbers as well as mathematical notations.
CRSUR 1.1	The notations should be clear and familiar to the users.
CRSF **2.0**	The system should recognize characters with noises as well.
CRSNF 2.1	The system should recognize the characters in less than 2 seconds.
HCSF **1.0**	The system should recognize hand written characters.
HCSNF 1.1	The system must have handwritten accuracy over 80%.
HCSUR 1.1	
IPSF **1.0**	The system should be able to capture image to perform calculations.
IPSNF 1.1	The system should use mobile camera to capture the image.
IPSNF 1.2	The system should capture the in 720*720 resolution.
IPSNF 1.3	After capturing the image there should be crop option.
IPSNF 1.4	There should be save option for image.
IPSUR 1.1	There should be camera icon to switch between standard and scanning calculator.
IPSF **2.0**	The system should be able to perform calculations on images from gallery.
IPSNF 2.1	While importing, the images should be shown from latest to old.
IPSNF 2.2	While importing only one image should be selectable.
IPSUR 2.1	The selected image should be faded.

5.4. System Modeling

5.4.1. Context Modeling

5.4.1.1. Use Case Diagram

- Use Case Diagram for Basic Calculation System

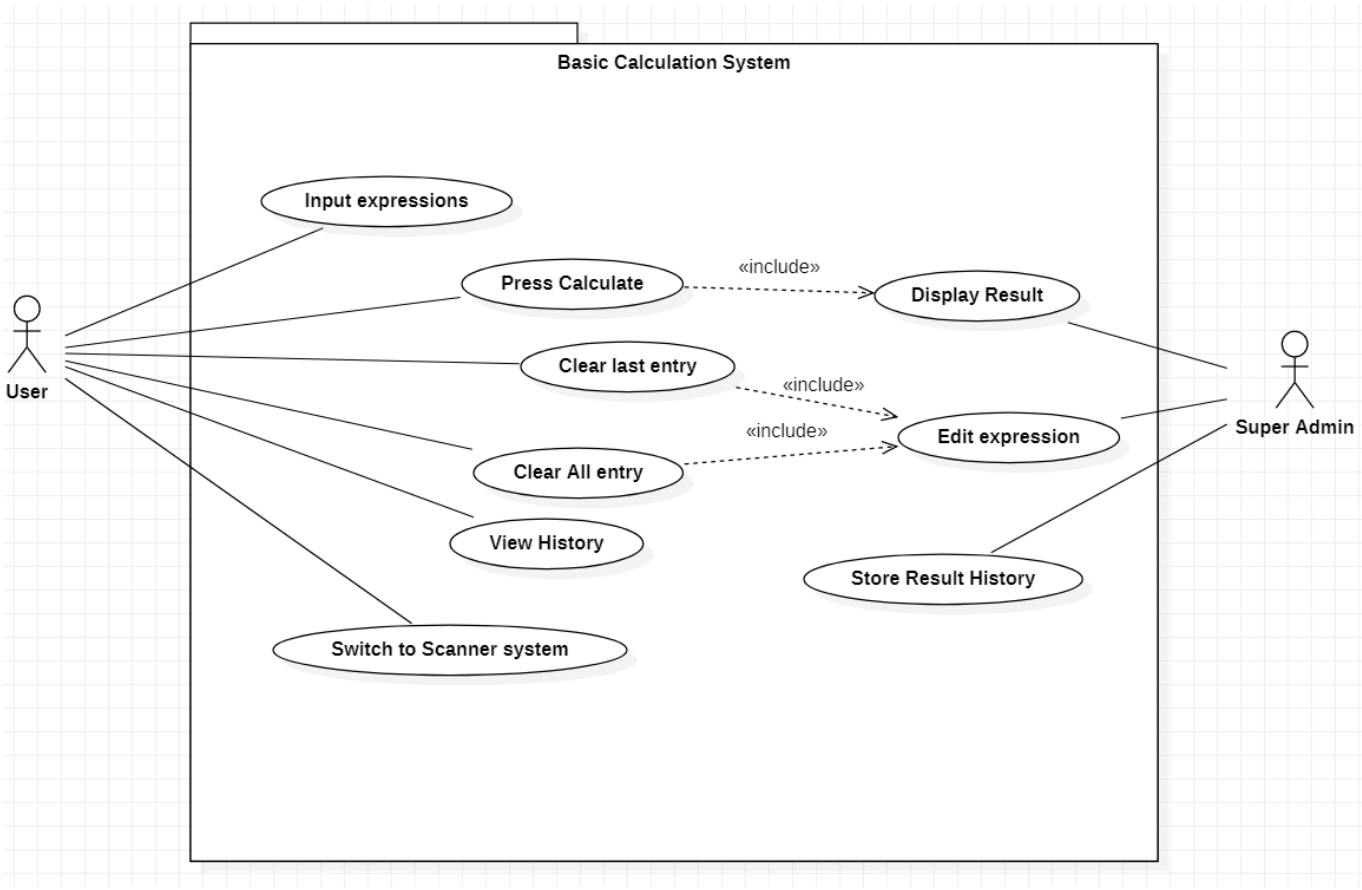


Figure 10: Use Case Diagram for Basic Calculation System

- Use Case Diagram for Image Capturing System

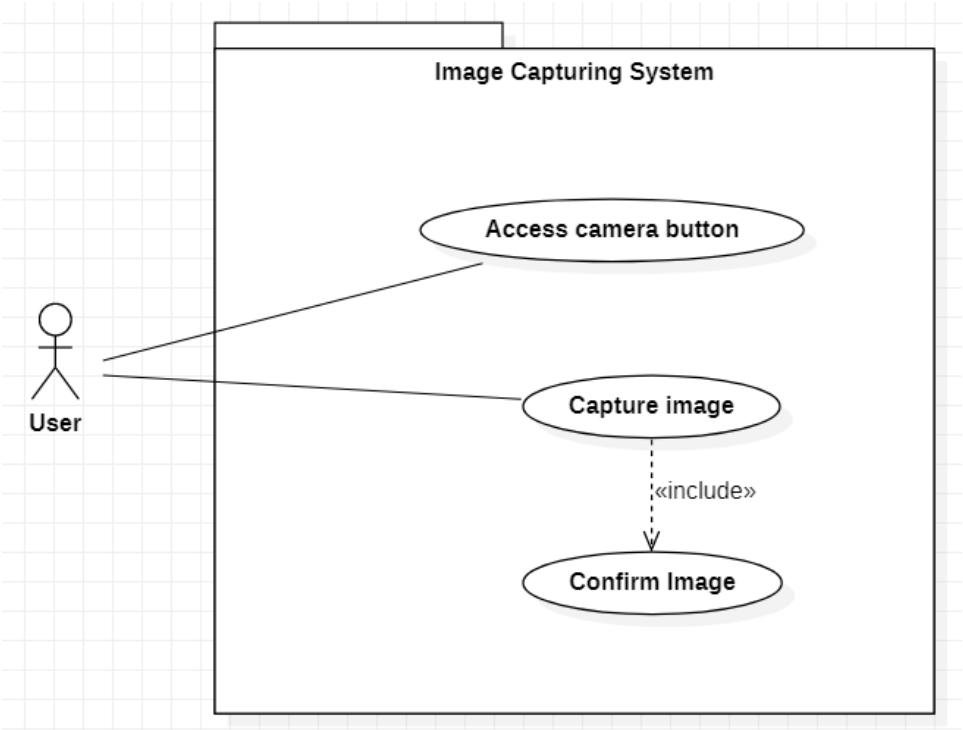


Figure 11: Use Case Diagram for Image Capturing System

- Use Case Diagram for Character Recognition System

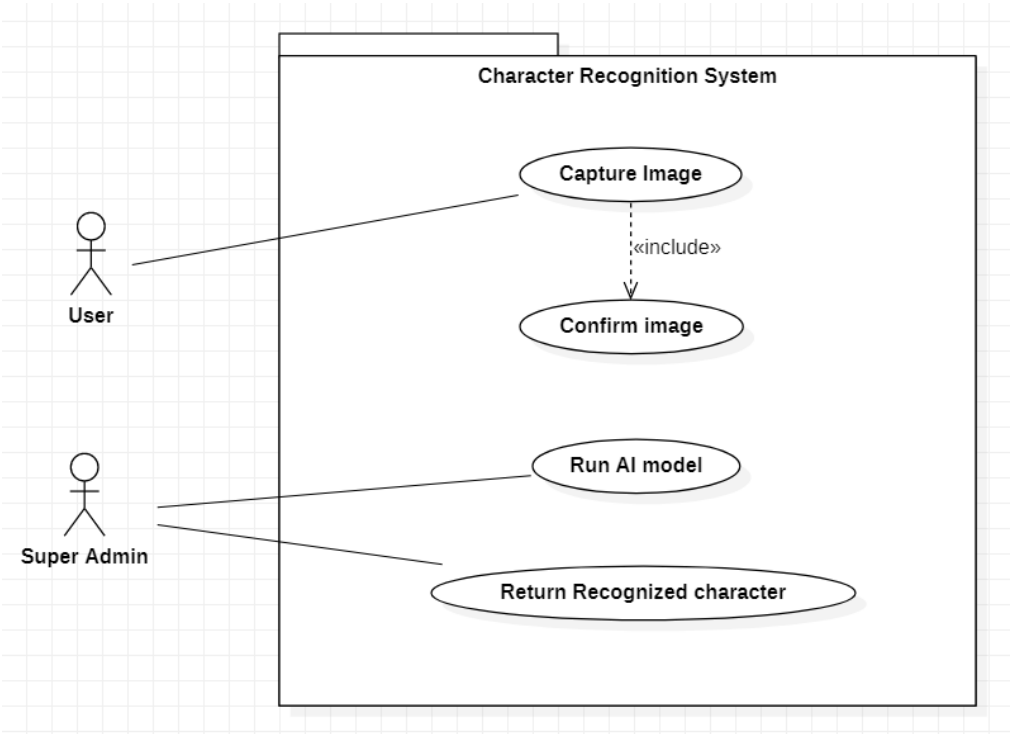


Figure 12: Use Case Diagram for Character Recognition System

5.4.2. Structural Modeling

5.4.2.1. Class Diagram

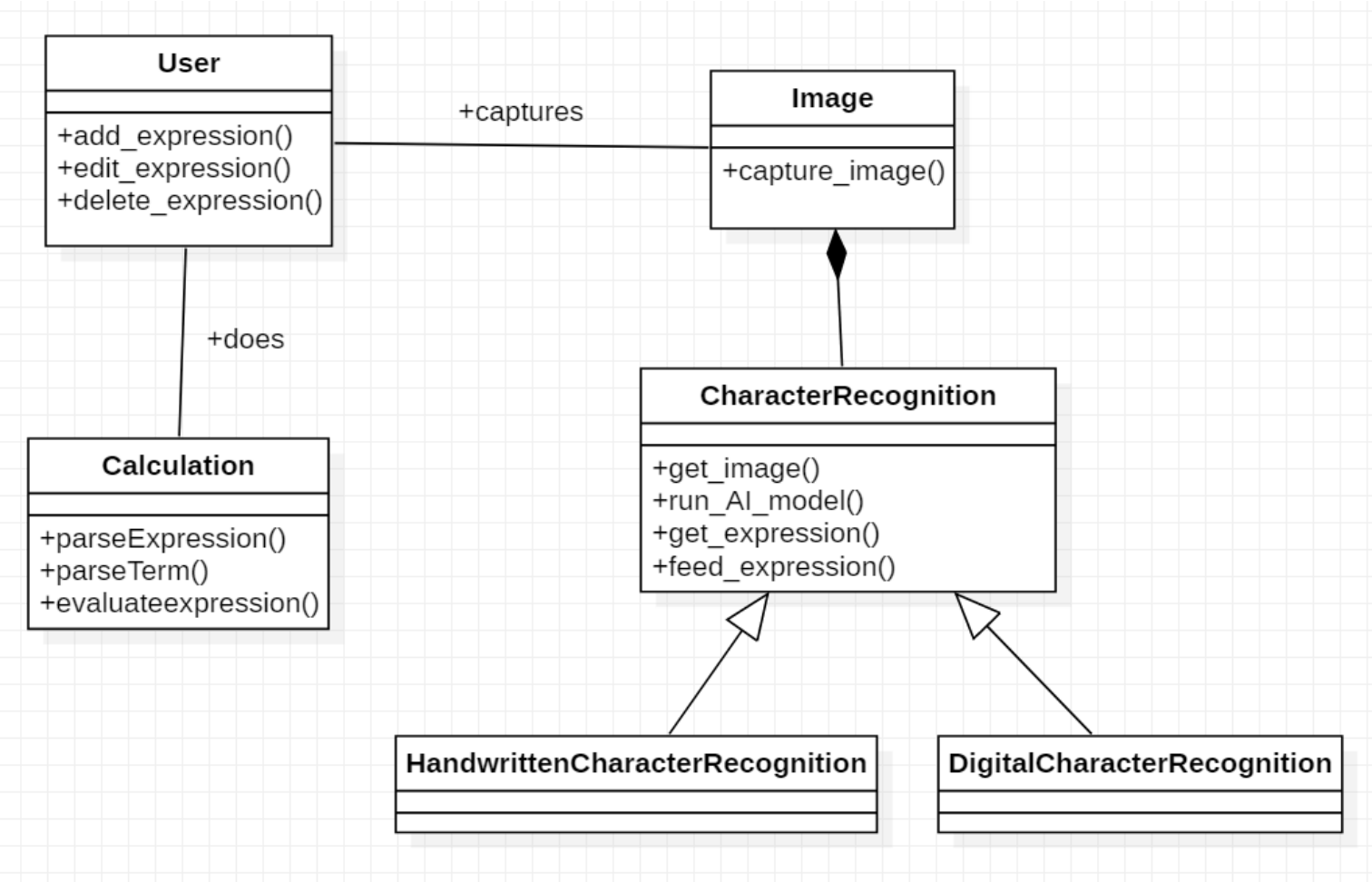


Figure 13: Class Diagram

5.4.3. Process Modeling

5.4.3.1. Context Diagram

- Context Diagram for Character Recognition System

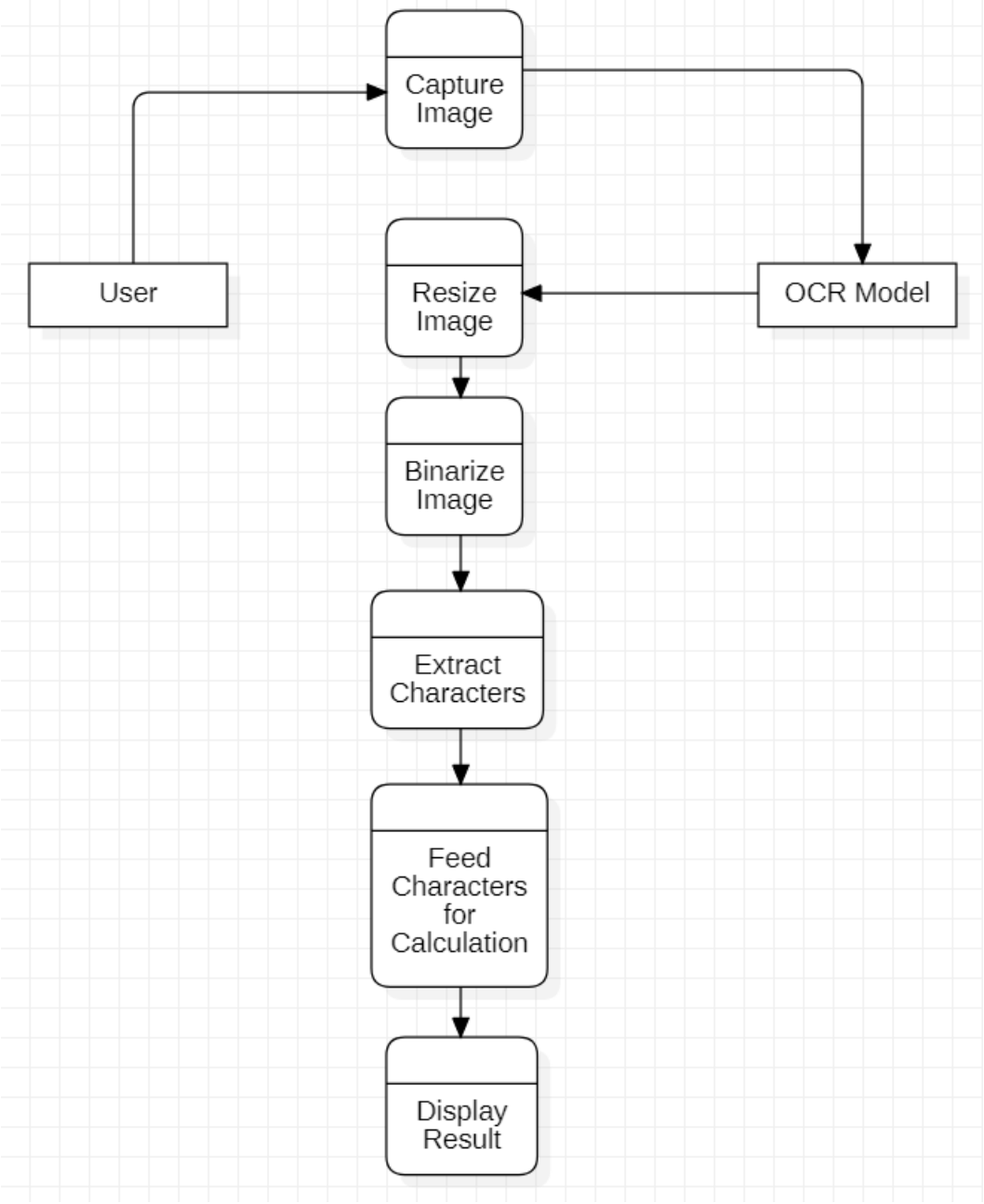


Figure 14: Context Diagram for Image Capturing System

- Context Diagram for Character Recognition System

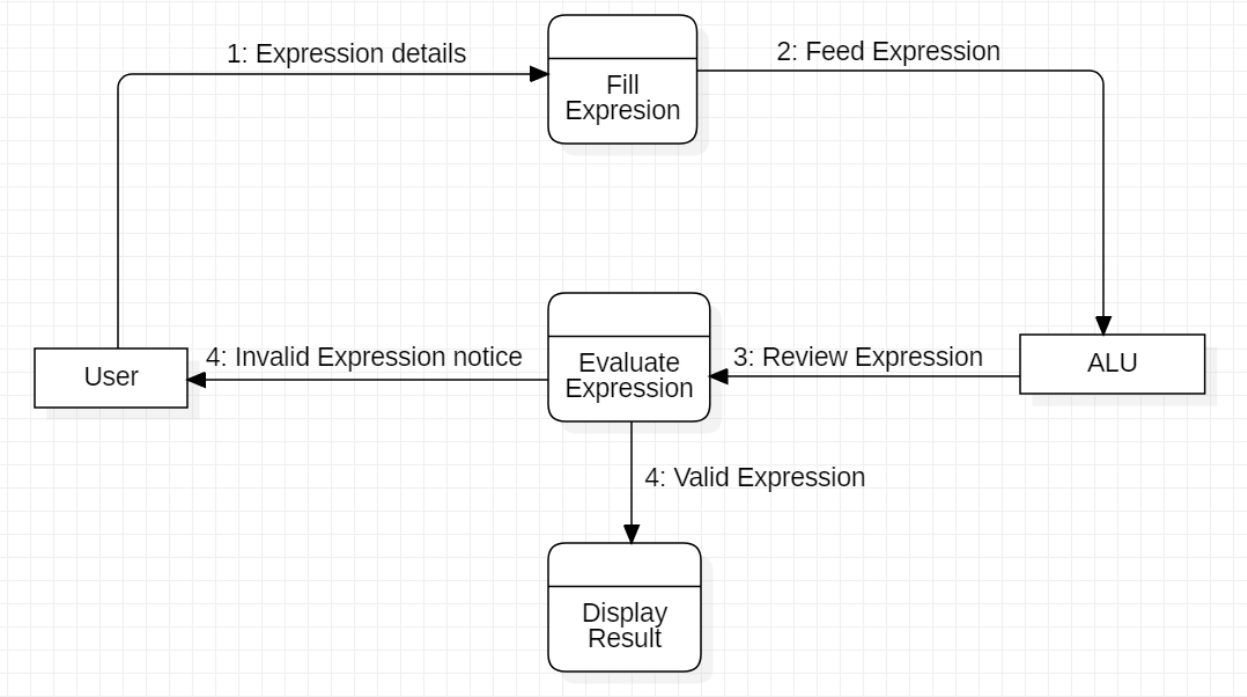


Figure 15: Context Diagram for Character Recognition System

5.4.4. UI Model

5.4.4.1. Wireframe

Figure 20: Main Screen

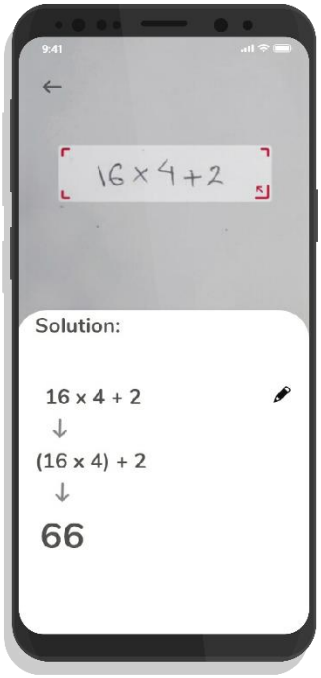


Figure 18: Solution Screen

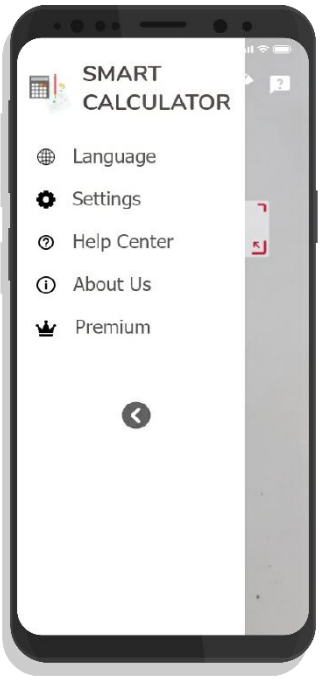


Figure 19: Navigation Screen

Figure 17: Calculator Screen

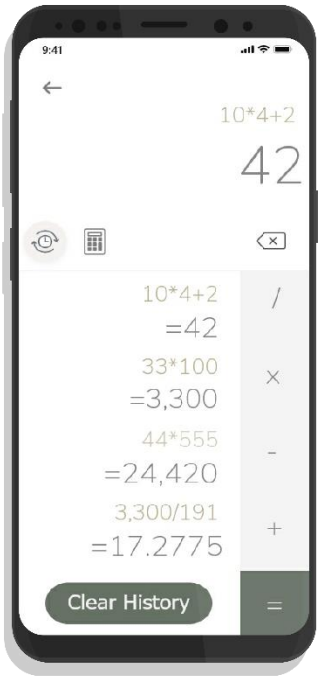


Figure 16: History Screen

6. Full details of Artefact

6.1. Development Methodology

Software development methodology is a series of processes or tasks used in software development in order to improve quality of a product. It can also be known as system development life cycle. Following certain methodology allows us to document policies, procedures and processes making development process easier.

As most of the development works happens in a team, so there needs to be a good communication between developers. This is where a methodology followed comes in handy. It sets norms between a group of people working on a project about how they are going to pass information between each-others. Whether that be documentation, discussion, or diagrams on paper. (Gianpaul Rachiele, 2018)

Some of the Software Development Methodologies are:

- Agile
- Waterfall
- Incremental
- Extreme Programming
- Rapid Application Development

Among these, Incremental model is best suited for my project and is defined below in section 6.2.

6.2. Preferred Methodology

For this software development Increment methodology is best suited.

Incremental Model is one of the methodologies of software development where requirements are divided into multiple standalone modules. Each of this modules go through phases involved in this methodology.

In this model a simple working system with basic features is built and delivered to client at first. Then the other features are added in many successive iterations. Due to this reason, this model is also known as Successive Version Model. (GeeksforGeeks, 2020)

In the below figure, Software Product is incrementally developed and delivered.

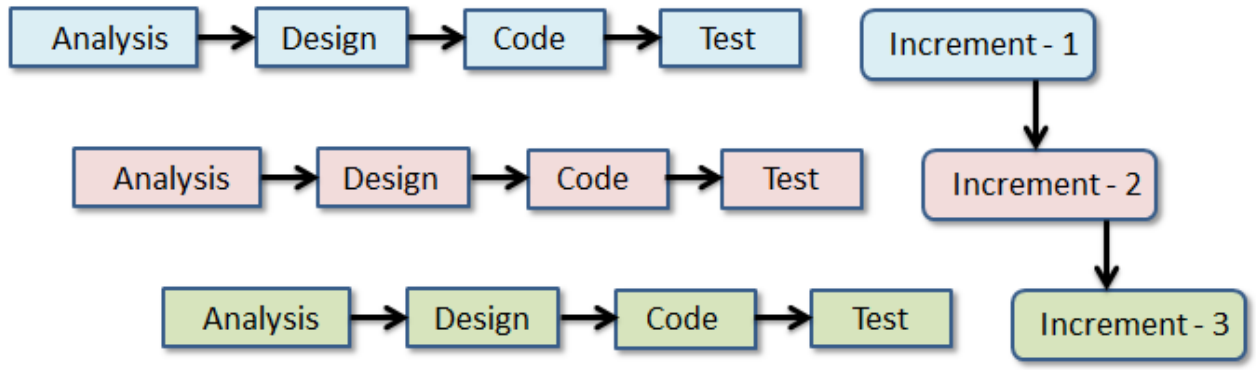


Figure 21: Incremental Model Visualization (EduCba, 2020)

Reasons to choose Incremental:

- Generates working software quickly and early during the software life cycle.
- Works on iteration, so it would be easy to develop promotional website at first and then mobile application.
- Parallel development can be done
- Supports change of requirements
- Testing and debugging is easier because of small iterations
- More flexible
- Easy to manage risks

6.3. Techniques and Tools

Software Development techniques includes preparing a plan of upcoming work, estimating task to perform, allocating resources and monitoring quality and deadlines. There are various techniques available for various development methodologies. Some of them are:

- Strategic Planning Technique
- Structured Programming Technique
- Object Oriented Analysis and Design Technique
- Software Testing Technique (Spring Digital, 2019)

From all the above mentioned techniques, Object Oriented Analysis and Design techniques is most suited for this project as the core programming language to be used in this project is *Python*, which is Object Oriented programming language. Similarly, for the front end part *Java* will be used which is also based on OOP concept.

When applying any techniques to the project we need to use specific tools for successfully implementing techniques. We needs tools to plan, design and develop the system. Some of the tools that come handy during software development are as follows:

- Modeling tools for designing
    - *Star UML, Adobe XD, Corel Draw*
- Word processor for report
    - *Microsoft Office*, Microsoft Excel
- Core Programming Language
    - *Python with Tesseract and OpenCV modules*
- Code Editor
    - *Visual Studio Code, Jupyter Notebook*
- Version Controlling
    - *Github*

6.4. Testing

As we are doing this project on Incremental fashion, each increment will give us simple working system with basic features. So, we can perform testing after every increment. And for the testing we will only conduct *Black Box Testing.*

6.5. Probable Issues During the project

- Hardware failure

One of the major issue that may occur during the project is hardware failure. No matter how careful we are, hardware like hard drive may crash at any time. Likewise problem in different peripherals like mouse and keyboard may occur.

To mitigate this problem project will be stored in cloud storage as well as eternal hardware. Extra pair of peripherals will also be available to use during the project.

- Version Controlling

As I am a beginner programmer, I have no experience to manage the flow of programming. While adding new features I might mess up the previous code and project may fail.

To overcome this issue I will use version controlling tools like “Github”. This way I can fall back to the previous running code.

- Lack of Dataset to train the system

Artificial Intelligence works on the basis of data provided to it. Similarly OCR will only recognize the data that is already fed to it. So we need huge amount of dataset to teach the system which may not be available free to use. And as a student I might be able to purchase those datasets.

To mitigate this problem we can ask for the data set that is already used by the senior students who had done similar type of projects.

- Incompatible SDLC methodology

Due to the insufficient experience in using different software development methodologies, the methodology chosen might not be compatible.

To mitigate this problem supervisor plays vital role. His guidance will help to choose the best methodology before starting the project. Moreover internship in similar project related companies will also help to build some experiences.

6.6. Plan/Schedule

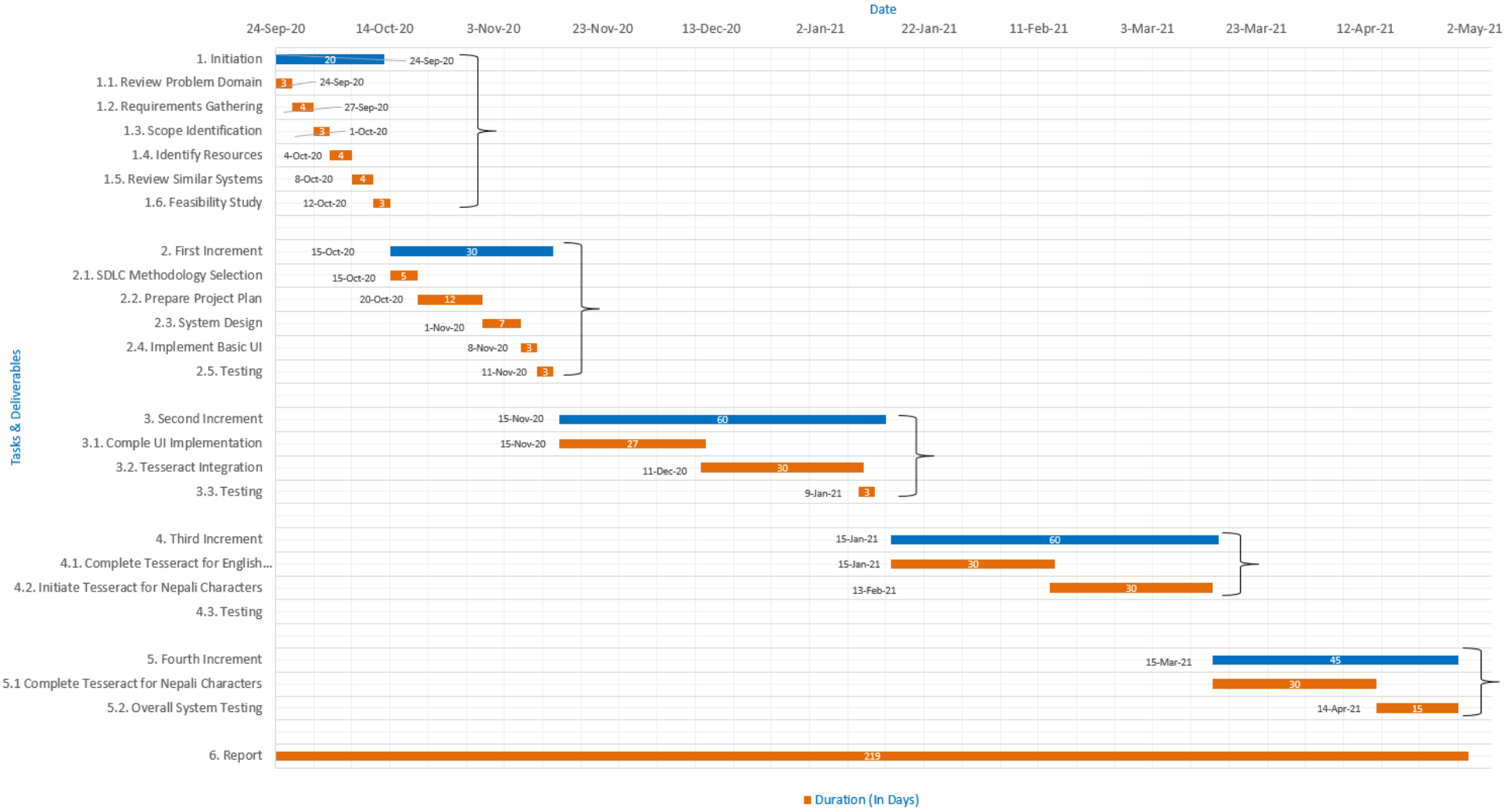


Figure 22: Gantt-Chart

7. Model Development

- 7.1. Data Collection
- 7.2. Data Training
- 7.3. Algorithm Comparison
- 7.4. Performance Analysis
- 7.5. Confusion Matrix
- 7.6. ROC Curve

References

CERN, Geneva, 1996. *19th CERN School of Computing.* Geneva, s.n.

EduCba, 2020. *Incremental Model.* [Online]
Available at: https://www.educba.com/incremental-model/
[Accessed 20 September 2020].

GeeksforGeeks, 2020. *GeeksfoGeeks.* [Online]
Available at: https://www.geeksforgeeks.org/software-engineering-incremental-process-model/
[Accessed 4 May 2020].

Gianpaul Rachiele, 2018. *Medium.* [Online]
Available at: https://medium.com/@gianpaul.r/software-development-methodologies-a856883a7630
[Accessed 5 May 2020].

Google Open Source, 2020. *Tesseract OCR.* [Online]
Available at: https://opensource.google/projects/tesseract
[Accessed 15 May 2020].

Khan, A. I. & Al-Habsi, S., 2019. *Machine Learning in Computer Vision.* Muscat, Remote Sensing and GIS Research Center.

Mathway, 2020. *Mathway.* [Online]
Available at: https://www.mathway.com/
[Accessed 08 October 2020].

Microsoft, 2019. *Microsoft Math Solver.* [Online]
Available at: https://math.microsoft.com/en
[Accessed 20 April 2020].

Patel, C. I., Patel, A. & Patel, D., 2012. Optical Character Recognition by Open source OCR Tool Tesseract. *International Journal of Computer Applications,* 55(10), pp. 50-56.

Patel, C. I. & Patel, D., 2012. Optical Character Recognition by Open Source OCR Tool Tesseract: A Case Study. *International Journal of Computer Applications,* 55(10), pp. 50-56.

Photomath, 2020. *Photomath.* [Online]
Available at: https://photomath.net/en/help/how-does-photomath-work
[Accessed 20 April 2020].

Ranjay, K., 2017. *Computer Vision: Foundations and Applications,* s.l.: Stanford University.

Sebe, N., Cohen, I., Garg, A. & Huang, T. S., 2005. *Machine Learning in Computer Vision.* 1st ed. New York: Springer.

Shodh Ganga, 2015. *Shodhganga:a reservoir of Indian theses.* [Online]
Available at: http://14.139.13.47:8080/jspui/handle/10603/130552
[Accessed 20 April 2020].

Sikka, A. & Wu, B., 2012. *Camera Based Equation Solver for Android Devices,* California: Stanford University.

Spring Digital, 2019. *Spring Digital.* [Online]
Available at: https://www.springdigital.com.au/software-apps/tools-and-techniques-for-software-development/
[Accessed 16 May 2020].

Wu, J., 2017. *Introduction to Convolutional Neural Networks,* Nanjing: National Key Lab for Novel Software Technology.