Intelligent Character Reader Using Artificial Neural Networks

Submitted by

Dipayan Deb-13000113032

Drona Banerjee-13000113033

Payal Kumari-13000113060

Karan Jaiswal-13000113039

Under the guidance of Prof. Poulami Dutta

Submitted for the partial fulfilment for the degree of Bachelor of Technology in Computer Science and Engineering

Techno India
EM 4/1, Salt Lake, Sector – V, Kolkata – 700 091.

CERTIFICATE

This is to certify that the project entitled "**Intelligent Character Reader Using Artificial Neural Networks (IANN)**", prepared by Dipayan Deb (13000113032), Drona Banerjee (13000113033), Payal Kumari (13000113060) and Karan Jaiswal (13000113039) of B.Tech (Computer Science & Engg.), Final Year, has been done according to the regulations of the Degree of Bachelor of Technology in Computer Science & Engineering. The candidates have fulfilled the requirements for the submission of the project report.

It is to be understood that, the undersigned does not necessarily endorse any statement made, opinion expressed or conclusion drawn thereof, but approves the report only for the purpose for which it has been submitted.

Ms. Poulami Dutta
(Internal Guide)
In Charge & Asst. Professor
Dept. of CS

Ms. Poulami Dutta
In Charge & Asst. Professor
Dept. of CSE

(Signature of External Examiner
with Designation and Institute)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

ACKNOWLEDGEMENT

We would like to express our sincere gratitude to Prof. Poulami Dutta of the department of Computer Science and Engineering, whose role as project guide was invaluable for the project. We are extremely thankful for the keen interest she took in advising us, for the books and reference materials provided for the moral support extended to us.

Last but not the least we convey our gratitude to all the teachers for providing us the technical skill that will always remain as our asset and to all non-teaching staff for the gracious hospitality they offered us.

Place: Techno India, Salt Lake

Date: 25/05/2017

……
Dipayan Deb

…….....
Drona Banerjee

…….
Karan Jaiswal

…….
Payal Kumari

Contents: **Page Numbers**

List of tables: **Page Numbers**

List of figures: **Page Numbers**

1 INTRODUCTION

1.1 BRIEFING

Intelligent Character Reader (ICR) is a software tool that enables us to convert our handwritten or printed texts into digital, searchable text formats by using self-learning algorithms which learn to recognize human handwriting in a way which is similar to how humans learn new concepts. It is an advanced Machine Learning application which automatically updates its recognition database for new handwriting patterns using its self-learning system, known as Neural Networks. When fed with input data in the form of an image file such as .PNG, .JPEG or any other image file formats, the system is expected to process the image, identify the texts in the image, segment the characters, classify the characters in accordance with the existing database and then produce an output in a searchable text format like .DOC, .PDF, etc. ICRs can be used for several other purposes like automated forms processing, letter sorting, etc.

1.2 PROBLEM DOMAIN

Intelligent character recognition (ICR) is an advanced optical character recognition (OCR) or rather more specific — handwriting recognition system that allows fonts and different styles of handwriting to be learned by a computer during processing to improve accuracy and recognition levels.
Most ICR software has a self-learning system referred to as a **neural network**, which automatically updates the recognition database for new handwriting patterns

Artificial Neural Network - In **machine learning**, **artificial neural networks**

(**ANNs**) is a network inspired by biological neural networks (the central nervous systems of animals, in particular the brain) which are used to estimate or approximate functions that can depend on a large number of inputs that are generally unknown.

Machine Learning-Machine learning is a subfield of computer science(more particularly soft computing) that evolved from the study of pattern recognition and computational learning theory in artificial intelligence. It's the field of study that gives computers the ability to learn without being explicitly programmed. Machine learning explores the study and construction of algorithms that can be trained and can perform predictive analysis on data.

1.3 RELATED STUDIES

A Pattern is a set of objects or phenomenon or concepts where the elements of sets are similar to one another in certain ways or aspects [1]. Pattern Recognition is one of important and vast fields of computer intelligence, which focuses on the recognition of patterns and regularities of data. It is an act of taking input of raw data and extract specific or unique features to recognize that data. 1985(Watanabe) said that pattern recognition can be looked as categorization problem, as inductive process, as structure analysis, as discrimination method and so on [2]. Optical Character Recognition is an application of pattern recognition. OCR is a method of converting a scanned image of typewritten, handwritten or printed text into computer readable format [3]-[5] i.e. a format that is understood by machine. Now when a page is scanned, it is stored in TIF format. When an image is displayed on the monitor, humans can read it, but to a computer, it's just a sequence of black and white pixels. Computer does not recognize any word in image. The main aim of OCR is to look at the image and try to decide if these pixels are representing a particular letter or not.

An Artificial Neuron is basically an engineering approach of biological neuron [6]-[7] i.e. Neural Networks basically aim at mimicking the structure and functioning of the human brain, to create intelligent behaviour. A Neural Network is a massively parallel distributed processor made up of simple processing units which have natural propensity for storing experiential knowledge and making it available for use. It resembles to brain in two aspects. First, Knowledge is acquired by the Network from its environment through a learning process.

Second, Interneuron connection strength is used to store acquired knowledge. In Neural Network, each node performs some simple computation and each connection conveys a signal from one node to another labelled by a number called ―connection strength [8].

Learning is formally defined as a process by which free parameters of a Neural Networks are adapted through a process of simulation by the environment in which the network is embedded. Once the system begins to learn containing some initial weight values, as the learning process increase weight values keeps on changing and provide the final output at end. Learning can be classified as: First, Supervised Learning i.e. learning with Teacher, Second, Unsupervised Learning i.e. learning without Teacher. Typical pattern recognition systems are designed using two pass. The first pass is a feature extractor that finds features within the data which are specific to the task being solved. The second pass is the classifier, which is more general purpose and can be trained using a neural network and sample data sets.

As Optical Character Recognition is defined as a Multiclass Problem, amongst various classification methods [9]-[10], we have used, Multilayer Feed Forward Architecture, which contains an Input Layer, an Output Layer and one or more Hidden Layer [11]. As the number of Hidden Layer increases the complexity of network also increases. Back-Propagation Neural Network (BPNN) algorithm is the most popular and the oldest supervised learning multilayer feed-forward neural network algorithm proposed by Rumelhart, Hinton and Williams [12]-[14]. Input vectors and the corresponding target vectors are used to train a network until it can approximate a function, associate input vectors with specific output vectors, or classify input vectors in an appropriate way as defined by you. Networks with biases, a sigmoid layer, and a linear output layer are capable of approximating any function with a finite number of discontinuities.

1.4 GLOSSARY

Table 1: Glossary

DEFINITION	*TERM*
The image format stores an image as 2D matrix. Each Pixel is represented by a bit Integer. Therefore highest value being 255(White) and lowest being 0(Black).	Greyscale Image
Transforming digital information representing images.	Image Processing
Image segmentation is a process of dividing an image into multiple parts. This is typically used to identify objects or other relevant information in digital images.	Segmentation
Machine Learning Algorithm uses a learning algorithm to learn from a dataset of correct solutions.	MNIST Dataset

2 PROBLEM DEFINITION

2.1 SCOPE

The purpose of this project is to take handwritten or printed English characters as Input, process the character, train the neural network algorithm, to recognize the pattern and modify the character to a searchable, digital version of the input. This project is aimed at developing software which will be helpful in recognizing characters, numeric as well as Special Characters of English language. This project is not restricted to English characters only. The design of the project is such that by adding a trained module in any other language is enough to add support for the language characters. One of the primary means by which

computers are endowed with human like abilities is through the use of neural network. Neural Networks are particularly for solving problems that cannot be expressed as a series of steps such as recognizing patterns classifying them into groups, series prediction and data mining.

2.2 EXCLUSION

The significant exclusions in this project are as follows:

- The software does not use Natural Language Processing (NLP) and is thus not capable of making sense out of sentences or words. Hence, predictive correction of errors based on meaning of a sentence is not possible.
- The system should process the input given by the user only if it is an image file (.jpg).

2.3 ASSUMPTIONS

- The quality of the camera used must be good.

3 PROJECT PLANNING

3.1 SOFTWARE LIFE CYCLE MODEL

For our project, we will be using the Iterative Waterfall Model.
An iterative life cycle model does not attempt to start with a full specification of requirements. Instead, development begins by specifying and implementing just part of the software, which can then be reviewed in order to identify further requirements. This process is then repeated, producing a new version of the software for each cycle of the model.

The iterative model can be broadly classified into four phases which unlike the classical waterfall model, are totally intertwined:

- Requirement Analysis and Specification: In this phase of the cycle the different requirements gathered are as follows:
    1) Functional requirements – Recognition of handwritten or printed characters including special characters
    2) Non-functional requirements – Recognizing the characters as fast as possible in an efficient manner.
    3) Goal of implementation– To create efficient and effective software for Handwriting recognition.

    In this phase the requirements for the software are gathered and analyzed. Iteration eventually resulted in a requirements phase that produces a complete and final specification of requirements.

- Feasibility Study: Analysis of the problem and collection of all the relevant information relating to the project was done. These collected data are analyzed to arrive at the following:
    1) An abstract problem definition
    2) Formulating different strategies for solving the problem like Image recognition by comparing pixels or Image recognition using Neural networks
    3) Feasibility, benefits and shortcomings of the strategies were evaluated.

- Design: Data Flow Diagram are created as shown in Appendix A according to the Requirement gathered in the previous phase of this life cycle.

- Coding and Testing: The software design is translated into source code in this phase. The final product is broken into number of modules like testing the Alphabets recognition, Numeric recognition and Special character recognition and each module is tested in isolation and then brought together to be tested as complete system. The machine learning module of the software was coded using Octave and the Graphical User Interface (GUI) has been programmed with Java.

- Review: In this phase, the software is evaluated, the current requirements

are reviewed, and changes and additions to requirements are proposed.

For each cycle of the model, a decision was made as to whether the software produced by the cycle will be discarded, or kept as a starting point for the next cycle. Eventually a point was reached where the requirements were complete and the software could be delivered, or it would have been impossible to enhance the software as required, and a fresh start would have been made.

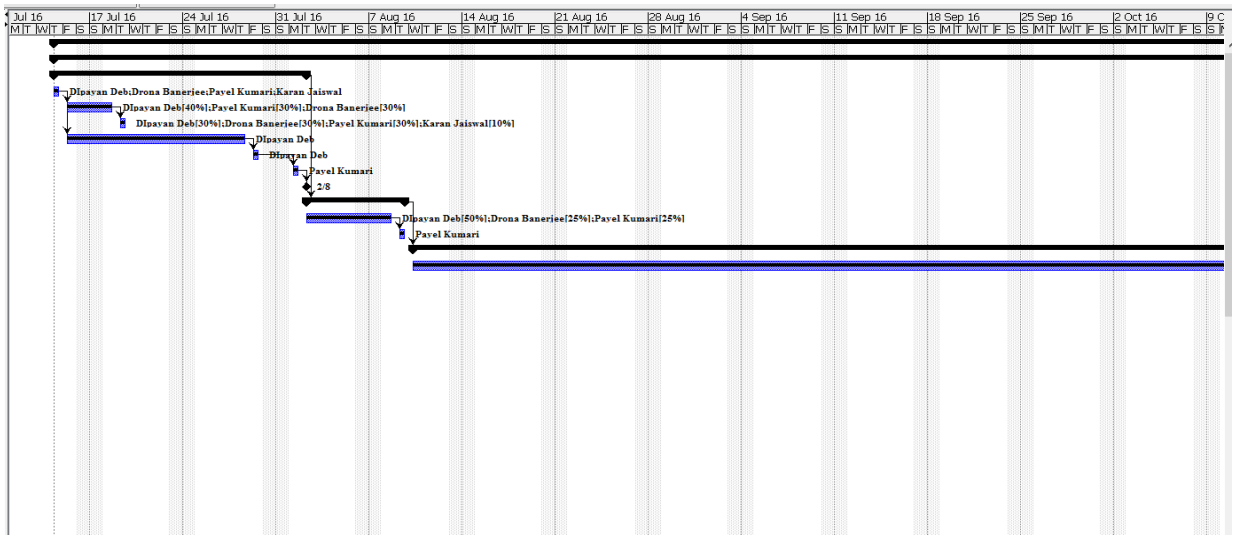
The iterative life cycle model can be likened to producing software by successive approximation. Drawing an analogy with mathematical methods that use successive approximation to arrive at a final solution, the benefit of such methods depends on how rapidly they converge on a solution.

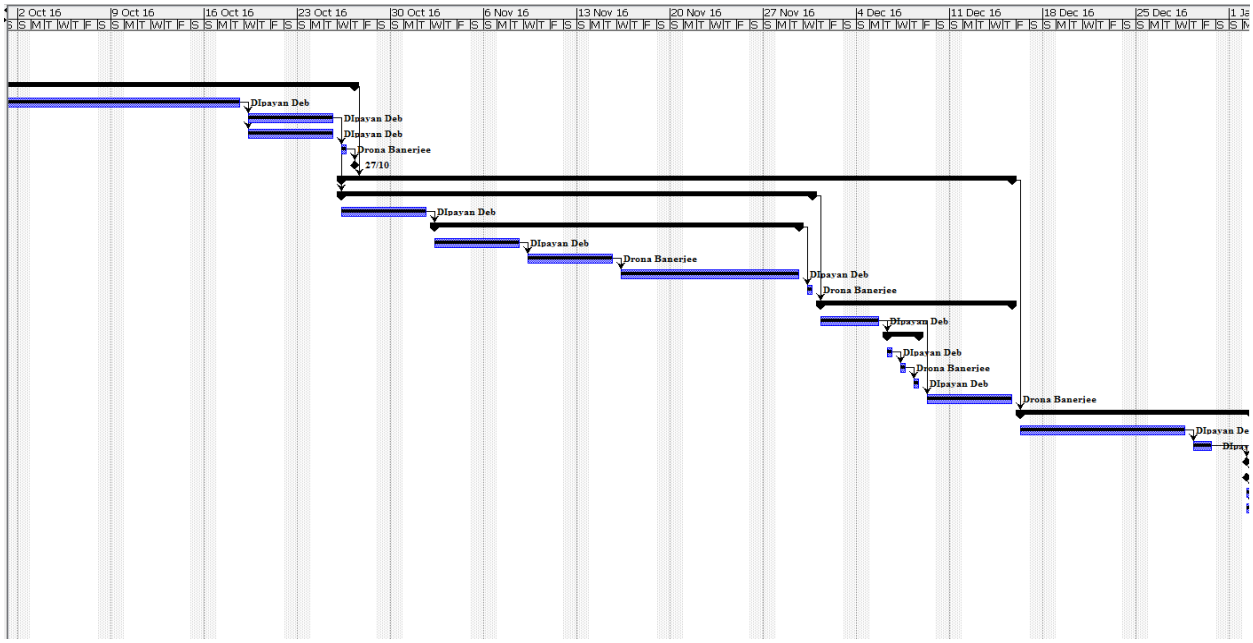
The key to successful use of an iterative software development life cycle has been rigorous validation of requirements, and verification (including testing) of each version of the software against those requirements within each cycle of the model.

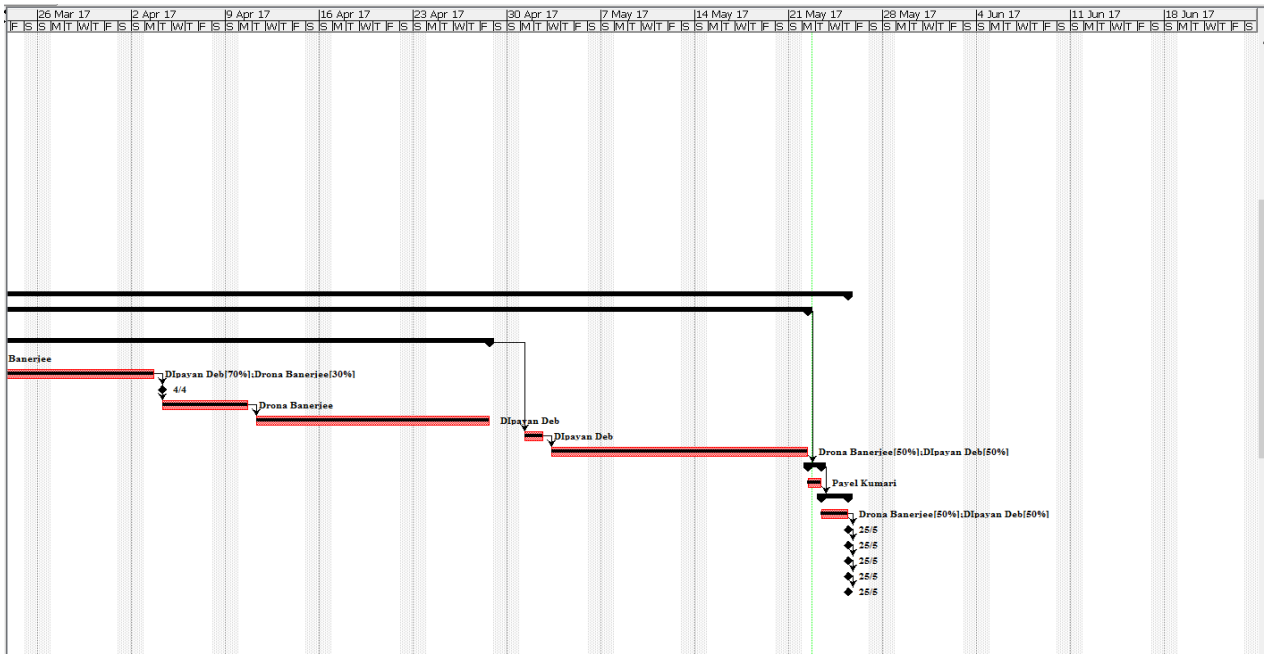
3.2 SCHEDULING

	Name	Duration	Start	Finish	Predecessors	Resource Names
1	**CSE Academic Project**	232.333 ...	**14/7/16 8:00 AM**	**5/6/17 10:40 AM**		
2	**Phase 1: 7th Semester Activities**	123 days	**14/7/16 8:00 AM**	**2/1/17 5:00 PM**		
3	**Project Startup**	13 days	**14/7/16 8:00 AM**	**2/8/16 8:00 AM**		
4	Team Building	1 day	14/7/16 8:00 AM	14/7/16 5:00 PM		DIpayan Deb;Drona Ba...
5	Brainstorm on Project Topic	2 days	15/7/16 8:00 AM	18/7/16 5:00 PM	4	DIpayan Deb[40%];Pay...
6	Project agreed with guide	1 day	19/7/16 8:00 AM	19/7/16 5:00 PM	5	DIpayan Deb[30%];Dro...
7	Related Study & Doumentation	10 days	15/7/16 8:00 AM	28/7/16 5:00 PM	4	DIpayan Deb
8	Deliver Project Synopsys for Guide's review	1 day	29/7/16 8:00 AM	29/7/16 5:00 PM	7	DIpayan Deb
9	Close review feedbacks	1 day	1/8/16 8:00 AM	1/8/16 5:00 PM	8	Payel Kumari
10	Project Synopsys Finalized	0 days	2/8/16 8:00 AM	2/8/16 8:00 AM	9	DIpayan Deb[50%];Pay...
11	**Requirement Analysis**	6 days	**2/8/16 8:00 AM**	**9/8/16 5:00 PM**	**3**	
12	Gather Requirements	5 days	2/8/16 8:00 AM	8/8/16 5:00 PM		DIpayan Deb[50%];Dro...
13	Prepare Draft Requirement Matrix	1 day	9/8/16 8:00 AM	9/8/16 5:00 PM	12	Payel Kumari
14	**Elaborate Rqmt & Document**	56 days	**10/8/16 8:00 AM**	**27/10/16 8:00 AM**	**11**	
15	Artificial Neural Networks	50 days	10/8/16 8:00 AM	18/10/16 5:00 PM		DIpayan Deb
16	Java / Octave	5 days	19/10/16 8:00 AM	25/10/16 5:00 PM	15	DIpayan Deb
17	Character Recognition	5 days	19/10/16 8:00 AM	25/10/16 5:00 PM	15	DIpayan Deb
18	Update Requirement Matrix	1 day	26/10/16 8:00 AM	26/10/16 5:00 PM	16	Drona Banerjee
19	Requirement Matrix Finalized	0 days	27/10/16 8:00 AM	27/10/16 8:00 AM	18	Drona Banerjee
20	**Design**	37 days	**26/10/16 8:00 AM**	**15/12/16 5:00 PM**	**14**	
21	**Detailed Design**	26 days	**26/10/16 8:00 AM**	**30/11/16 5:00 PM**	**16**	
22	Image Processing	5 days	26/10/16 8:00 AM	1/11/16 5:00 PM		DIpayan Deb
23	**Machine learning module**	20 days	**2/11/16 8:00 AM**	**29/11/16 5:00 PM**	**22**	
24	Number Module	5 days	2/11/16 8:00 AM	8/11/16 5:00 PM		DIpayan Deb
25	Special character module	5 days	9/11/16 8:00 AM	15/11/16 5:00 PM	24	Drona Banerjee
26	Character module	10 days	16/11/16 8:00 AM	29/11/16 5:00 PM	25	DIpayan Deb
27	Training with MNIST	1 day	30/11/16 8:00 AM	30/11/16 5:00 PM	23	Drona Banerjee
28	**Test Plan Preparation**	11 days	**1/12/16 8:00 AM**	**15/12/16 5:00 PM**	**21**	
29	Preprocessing of Image	3 days	1/12/16 8:00 AM	5/12/16 5:00 PM		DIpayan Deb
30	**Machine learning**	3 days	**6/12/16 8:00 AM**	**8/12/16 5:00 PM**	**29**	
31	Number Module	1 day	6/12/16 8:00 AM	6/12/16 5:00 PM		DIpayan Deb
32	Special Character Module	1 day	7/12/16 8:00 AM	7/12/16 5:00 PM	31	Drona Banerjee
33	Character Module	1 day	8/12/16 8:00 AM	8/12/16 5:00 PM	32	DIpayan Deb
34	Output	5 days	9/12/16 8:00 AM	15/12/16 5:00 PM	29	Drona Banerjee
35	**Phase 1 Closure**	12 days	**16/12/16 8:00 AM**	**2/1/17 5:00 PM**	**20**	
36	Prepare 7th Semester Project Report	9 days	16/12/16 8:00 AM	28/12/16 5:00 PM		DIpayan Deb[40%];Dro...
37	Prototype 1	2 days	29/12/16 8:00 AM	30/12/16 5:00 PM	36	DIpayan Deb
38	Updated Requirement Matrix	0 days	2/1/17 8:00 AM	2/1/17 8:00 AM	37	Payel Kumari
39	Updated Project Plan	0 days	2/1/17 8:00 AM	2/1/17 8:00 AM	38	Drona Banerjee
40	Project Viva	1 day	2/1/17 8:00 AM	2/1/17 5:00 PM	39	
41	Approved Project Report - 7th Semester	1 day	2/1/17 8:00 AM	2/1/17 5:00 PM	39	
42	Semester Gap	12 days	20/1/17 8:00 AM	6/2/17 5:00 PM	2	

Semester Gap	12 days	20/1/17 8:00 AM	6/2/17 5:00 PM	2	
Phase 2: 8th Semester Activities	77.333 d...	**7/2/17 8:00 AM**	**25/5/17 10:40 AM**	**42**	
Coding & Únit Testing	74.333 d...	**7/2/17 8:00 AM**	**22/5/17 10:40 AM**		
Image Processing	20 days	7/2/17 8:00 AM	6/3/17 5:00 PM		DIpayan Deb
Machine learning	39 days	**7/3/17 8:00 AM**	**28/4/17 5:00 PM**	**45**	
Number Module	10 days	7/3/17 8:00 AM	20/3/17 5:00 PM		Drona Banerjee
Prototype-2 Neural Net Digit Classifier	10 days	21/3/17 8:00 AM	3/4/17 5:00 PM	47	DIpayan Deb[70%];Dro...
Checkpoint review	0 days	4/4/17 8:00 AM	4/4/17 8:00 AM	48	
Special Character Module	5 days	4/4/17 8:00 AM	10/4/17 5:00 PM	48	Drona Banerjee
Character Module	14 days	11/4/17 8:00 AM	28/4/17 5:00 PM	50	DIpayan Deb
Training with Dataset	2 days	1/5/17 8:00 AM	2/5/17 5:00 PM	46	DIpayan Deb
Character extraction using CNN	13.333 days	3/5/17 8:00 AM	22/5/17 10:40 AM	52	Drona Banerjee[50%];...
System Integration Testing	1 day	**22/5/17 10:40 AM**	**23/5/17 10:40 AM**	**44**	
Handwriting recognition Testing	1 day	22/5/17 10:40 AM	23/5/17 10:40 AM		Payel Kumari
Project Closure	2 days	**23/5/17 10:40 AM**	**25/5/17 10:40 AM**	**54**	
Prepare 8th Semester Project Report	2 days	23/5/17 10:40 AM	25/5/17 10:40 AM		Drona Banerjee[50%];...
Updated Requirement Matrix	0 days	25/5/17 10:40 AM	25/5/17 10:40 AM	57	DIpayan Deb
Updated Project Plan	0 days	25/5/17 10:40 AM	25/5/17 10:40 AM	58	Drona Banerjee
Review by Internal Faculty	0 days	25/5/17 10:40 AM	25/5/17 10:40 AM	59	
Review by Faculties	0 days	25/5/17 10:40 AM	25/5/17 10:40 AM	60	
Approved Project Report -8th Semester	0 days	25/5/17 10:40 AM	25/5/17 10:40 AM	61	


Jul 16
17 Jul 16
24 Jul 16
31 Jul 16
7 Aug 16
14 Aug 16
21 Aug 16
28 Aug 16
4 Sep 16
11 Sep 16
18 Sep 16
25 Sep 16
2 Oct 16
Dipayan Deb;Drona Banerjee;Payel Kumari;Karan Jaiswal
Dipayan Deb[40%];Payel Kumari[30%];Drona Banerjee[30%]
Dipayan Deb[30%];Drona Banerjee[30%];Payel Kumari[30%];Karan Jaiswal[10%]
Dipayan Deb
Dipayan Deb
Payel Kumari
2/8
Dipayan Deb[50%];Drona Banerjee[25%];Payel Kumari[25%]
Payel Kumari


2 Oct 16
9 Oct 16
16 Oct 16
23 Oct 16
30 Oct 16
6 Nov 16
13 Nov 16
20 Nov 16
27 Nov 16
4 Dec 16
11 Dec 16
18 Dec 16
25 Dec 16
Dipayan Deb
Dipayan Deb
Dipayan Deb
Drona Banerjee
27/10
Dipayan Deb
Dipayan Deb
Drona Banerjee
Dipayan Deb
Drona Banerjee
Dipayan Deb
Dipayan Deb
Drona Banerjee
Dipayan Deb
Drona Banerjee


26 Mar 17
2 Apr 17
9 Apr 17
16 Apr 17
23 Apr 17
30 Apr 17
7 May 17
14 May 17
21 May 17
28 May 17
4 Jun 17
11 Jun 17
18 Jun 17
Banerjee
DIpayan Deb[70%];Drona Banerjee[30%]
4/4
Drona Banerjee
DIpayan Deb
DIpayan Deb
Drona Banerjee[50%];DIpayan Deb[50%]
Payel Kumari
Drona Banerjee[50%];DIpayan Deb[50%]
25/5
25/5
25/5
25/5
25/5

3.3 COST ANALYSIS

We have used the basic COCOMO model for cost estimation.

Estimated KLOC= 2.7

As our project is Semi-Detached, a1= 3.0, a2=1.12, b1= 2.5 and b2= 0.35

Effort = $3.0 * (2.7)^{1.12} PM$
=9.13 PM.

Tdev = $2.5 * (9.13)\text{^}0.35\ Months$
= 5.42 Months

Assuming the labour cost is Rs10000 per month per employee

Size of team is 4

Therefore, cost of man power = Rs. (5.42*10000*4) =Rs. 216000

MATERIAL COST REQUIRED FOR SOFTWARE DEVELOPMENT:

Table 2: Cost Estimation

Cost(INR)	Quantity	Item
5999.00	1	Microsoft Office 2016 for Home and Student (Word, Excel, PowerPoint)
1900.00	1	Windows 7 starter edition 32bit

*The laptops or Personal computers are provided by the college to team members whose costing doesn't come into consideration.

Total Cost of materials: ₹7899.00

Therefore total cost is =Rs. (216000 + 7899) = Rs. 224699

4 REQUIREMENT ANALYSIS

4.1 REQUIREMENT MATRIX

Rqmt ID	*Requirement Item*	*Requirement Status*	*Design Module*	*Design Reference (section# under project Report)*	Test Case Number	*Technical Platform of Implementation*	*Prototype prepared ?*	Name of Program / Component	Test Results Reference	Additional Comments (if not included in previous columns)
INPUT	Image Input	Completed	0.4	5.3.1	T<5.3.2>	OCTAVE,Java	Yes	BlankFragment.java	ImageTest-1.txt	
IANN-1	Image Segmentation	Completed	0.1	5.3.2	T<5.3.2>	OCTAVE,Python,Java	Yes	BlankFragment.java	imageSegment.txt	
IANN-1.1	Line segmentation	Completed	0.1.1	5.3.2.1	T<5.3.2.1>	OCTAVE,Python,Java	Yes	BlankFragment.java	ImageSegment1.txt	
IANN-1.2	Word Segmentation	Completed	0.1.2	5.3.2.2	T<5.3.2.2>	OCTAVE,Python,Java	Yes	BlankFragment.java	ImageSegment2.txt	
IANN-1.3	Letter Segmentation	Completed	0.1.3	5.3.2.3	T<5.3.2.3>	OCTAVE,Python,java	Yes	BlankFragment.java	ImageSegment3.txt	
IANN-2	Image Processing	Completed	0.2	5.3.3	T<5.3.3>	OCTAVE,Python,java	Yes	OcrDectetorProcessor.jav	ImageText.txt	
IANN-2.1	Noise Removal	Completed	0.2.1	5.3.3.1	T<5.3.3.1>	OCTAVE,Python,java	Yes	OcrDectetorProcessor.jav	ImageText-2.txt	
IANN-2.2	Grayscale Convertion	Completed	0.2.2	5.3.3.2	T<5.3.3.2>	OCTAVE,Python,java	Yes	OcrDetectorProcessor.jav	ImageTest-1.txt	
IANN-2.3	1D vector convertion	Completed	0.2.3	5.3.3.3	T<5.3.3.3>	OCTAVE,Python,java	Yes	OcrDetectorProcessor.jav	ImageTest-3.txt	
IANN-3	Machine Learning	Completed	0.3	5.3.4	T<5.3.4>	OCTAVE,Python,java	Yes	BlankFragment.java	Accuracy.txt	
IANN-3.1	Number Detection Module	Completed	0.3.1	5.3.4.1	T<5.3.4.1>	OCTAVE,Python,java	YES	BlankFragment.java	DigitAccuracy.txt	
IANN-3.2	Character Detection Module	Completed	0.3.2	5.3.4.2	T<5.3.4.2>	OCTAVE,Python,java	Yes	BlankFragment.java	charAccuracy.txt	
IANN-3.3	Special Character Module	Completed	0.3.3	5.3.4.3	T<5.3.4.3>	OCTAVE,Python,java	Yes	BlankFragment.java	SplAccuracy.txt	
OUTPUT	Output Unit	Completed	0.5	5.3.5	T<5.3.5>	OCTAVE,Python,java	Yes	Text1.java	DigitAccuracy.txt	
GUI	Graphical User Interface	Completed	0.6	5.3.6	T<5.3.6>	JAVA(android)	Yes	MainActivity.java	GuiText.txt	

4.2 REQUIREMENT ELABORATION

4.2.1 Image Input

In this stage the software acquires an image as an input through a scanner or any suitable input method. This image should have a specific format such as .jpeg, .jpg, .png etc.

4.2.2 Image Segmentation

Convert any image into a series of Black Text written on a white background. Thus, it induces uniformity to all the input images. This also reduces computational power as it to deal with two colors i.e. Black and White.

4.2.2.1 Line Segmentation

Segmentation of a document image into its basic entities namely text lines and words, is a critical stage towards handwritten document recognition. The difficulties that arise in handwritten documents make the segmentation procedure a challenging task. There are many problems encountered in the segmentation procedure. For the text line segmentation procedure, these include the difference in the skew angle between lines on the page or even along the same text line, overlapping words and adjacent text lines touching.

4.2.2.2 Word Segmentation

Word Segmentation is an operation that seeks to decompose an image of a sequence of sentences into sub-images of individual words, separated by spaces. It is one of the decision processes in our system.

4.2.2.3 Letter Segmentation

Character segmentation or Letter Segmentation is an operation that seeks to decompose an image of a sequence of characters into sub-images of individual symbols. It is one of the decision processes in our system.

4.2.3 Image Processing

4.2.3.1 Noise Removal

Noise removal is a topic in document analysis that has been dealt with extensively for typed or machine-printed documents. For handwritten documents, the connectivity of strokes has to be preserved. Digital capture of images can introduce noise from scanning devices and transmission media. Smoothing operations are often used to eliminate the arti-crafts introduced during image

capture. One study, describes a method that performs selective and adaptive stroke “filling” with a neighborhood operator which emphasizes stroke connectivity, while at the same time, conservatively check aggressive “over-filling.

4.2.3.2 Greyscale conversion

In photography and computing, a greyscale or greyscale digital image is an image in which the value of each pixel is a single sample, that is, it carries only intensity information. Images of this sort, also known as black-and-white, are composed exclusively of shades of gray, varying from black at the weakest intensity to white at the strongest.

Greyscale images are distinct from one-bit bi-tonal black-and-white images, which in the context of computer imaging are images with only two colors, black and white (also called bi-level or binary images). Greyscale images have many shades of gray in between.

Our system requires converting the true color RGB image to the Greyscale intensity image.

4.2.3.3 1D Vector Conversion

Our system requires converting multi-dimensional array into 1D vector for proper processing.

4.2.4 Machine Learning

4.2.4.1 Number Detection Module

In this module the system is expected to detect numbers like 1,2,7,9,etc.

4.2.4.2 Character Detection Module

In this module the system is expected to detect regular characters like A, f, a, etc.

4.2.4.3 Special Character Module

In this module the system is expected to detect special characters like @, !, &, #, etc.

4.2.5 Output Unit

In this unit, the system is expected to produce the required output for the input image in a user-friendly manner. The output needs to be a text file having formats like .txt, .doc, .pdf, etc.

4.2.6 Graphical User Interface

To add to the intuitiveness of operation we need a Graphical User Interface which would provide users with immediate, visual feedback about the effect of each action. Commands are issued in the GUI by using a mouse, trackball or touchpad to first move a pointer on the screen to, or on top of, the icon, menu item or window of interest in order to *select* that object.

5 DESIGN

5.1 Technical Environment

For our projects, we will be using multiple programming languages.

1. Java - It is a cross platform language. This will be used to create the Graphical User Interface of the application,
2. Octave - This is a mathematical programming language. This will be used to implement the Machine Learning and Neural Network components of the project initially and test the workings.
3. Python - This is a very feature rich language in terms of Machine Learning and therefore can be used to further enhance the functionality of the project.

Table 3-The Specification

Hardware and Software	Specifications
Hard Disk	50MB (for final app),100GB (For Development)
RAM	1 GB, 8GB (For Development)
Processor	Qualcomm Processor (For App), Intel Core i5(For Development)
Operating System	Android (For App), Windows/Ubuntu(For Development)
Software Requirement	Android Studio, Java, Octave, Python3, Google Vision API ,TensorFlow

5.2 Hierarchy of Modules

The Hierarchy of the modules and sub-modules are as shown in the Level 0, Level 1 and Level 2 DFD diagram provided in appendix A.

The Level 0 Diagram shows that our program accepts a Image as a input and outputs the text written in the image.

The Level 1 Diagram goes into a bit of more details. It shows that before the machine learning algorithm can take on the input the Image must be segmented and processed.

The Level 2 Diagram shows the exact working and data flow in our project. It shows that first the segmenter segments all the lines into different images. After that it segments each of the lines into images of different words and those words get segmented to images of letters.
After that noise reduction is done to make sure that the input image is of highest quality. After that it gets converted into greyscale image. Further it gets converted into a 1D vector containing the intensities of each pixel. The number, special character and character sub-modules detect the respective output and then output it via the output unit.

5.3 Detailed Design

5.3.1 Input Module

This module takes a image as input which has text written in it. The text in the image will be extracted and stored in text form.

5.3.2 Segmentation

The input image consists of lines of text not just one character. This process will break down the image into images of the lines. Then it will further break down each line image into images of words. And finally each of those images of words will get broken into images of each letter of resolution 32*32*3.

5.3.3 Image Processing

The image of the letters needs to be processed before they can be used by the machine learning algorithm,

By default all the images are in RGB format. The RGB format is represented by a 3D matrix in the computer memory. The 3 2D matrices

consists of the intensity of colors red, green and blue pixel by pixel.

For your purpose, we will convert the Image into the greyscale format. This format is represented by a 2D array. We will remove all the necessary noise and then convert the values of the pixels between 0 and 1 for our algorithm to work. After that we will have to convert the 2D array into an 1D vector so that we can apply a logistic regression.

5.3.4 Machine Learning

5.3.4.1 Number Module

The number module will convert a number in a given image into text. The number module uses the Logistic Regression. This is being treated as a classification problem.
Logistic Regression uses the sigmoid function. The value of this function is between 0 and 1.
The number module will have 10 parts or sub modules. Each sub module will have code to detect respective number from 0 to 9 . The input will first go to the submodule that detects 1 and so on until it reaches the last sub module that detects 9. Each submodule will give the result based on the sigmoid function. The number represented by the submodule that gives the result closest to 1 will the required answer.

Sigmoid Function.

$g(z) = 1/(1 + e^{-z})$

$z=\theta^T x$

$h_\theta(x)=g(\theta^T x)$

θ^T- This is the transpose of the parameter vector. Each number has a different set of parameters.

x- This is a vector representing the 1D vector obtained from the input image.

The θ vector for a particular number is determined for a particular number with the help of algorithm called gradient descent.
A huge number of training set are taken and then provided to the gradient descent algorithm. This algorithm then uses the data set to properly the exact values of θ.

Gradient Descent
Repeat {

$\theta_j = \theta_j - \square((1/m) \sum_1^m (h_\theta(x^{(i)})-y^{(i)})x_j^{(i)})$
}

5.3.4.2 The special characters module

This module is used to detect the special characters. This works in exactly the same way as the numbers module. It utilizes the same algorithm. Just at the time of training a dataset of special characters is used instead of number dataset.

5.3.4.3 The characters module

This module will use a Neural Network to solve the problem. Each of the pixel intensities in the 1D array are used as inputs to the Neural Networks. We use a 3-layer Neural Network. The network consists of one hidden layer.

Shown above is a representation of a 3 layer neural networks. As we are using a 32*32*3 resolution Image as input there are 784 input neuron. Since there are 26 letters in the english language therefore there will be 26 output neurons. The above neural network has 1 hidden layer.

Similar to the previous algorithm the θ of the neural network are calculated using gradient descent algorithm with the help of backpropagation algorithm.

5.3.5 Output Unit

This module looks at the output of different machine learning module and then provides the desired output.

5.3.6 Graphical User Interface

To add to the intuitiveness of operation we need a Graphical User Interface which would provide users with immediate, visual feedback about the effect of each action. Commands are issued in the GUI by using a mouse, trackball or touchpad to first move a pointer on the screen to, or on top of, the icon, menu item or window of interest in order to *select* that object.

5.4 Test Planning

Table 4: Test Plan

Output	Input	Test ID	Serial Number
2D matrix form of the RGB image.	RGB image provided as input	T<5.3.1>	1
It will output a much clearer image	The 2d Matrix image is provided as input	T<5.3.3.1>	2
Convert it to grayscale before converting it to 2d Matrix format	Will Take RGB image as input	T<5.3.3.2>	3
Will convert to a 1d vector array for further processing	Will take in the noise removed imaged	T<5.3.3.3>	4
Give the output after processing through a Neural Network	Will Take the 1d vector	T<5.3.4.1>	5

6 IMPLEMENTATION

6.1 Implementation Details

6.1.1 Image Input

Uses CameraView of android to directly feed camera data to TextRecognizer class. For Gallery all the images are displayed in a GridView. Once an Images is clicked, it is converted into a Bitmap Image and Inputted into the TextRecognizer class for conversion.

6.1.2 Image Segmentation

In a particular image containing text. The space above and below the line is nothing but white space. Similarly, the same is between words and between letters there are specific patterns.

6.1.2.1 Line Segmentation

A Neural Network is trained how a line looks like. Then it starts from the top of the image and scans the whole image in search of lines.

6.1.2.2 Word Segmentation

The space between two letters contain spaces. We will use this to segment lines. A Neural Network is trained to look for spaces between letters similar to the process of searching lines.

6.1.2.3 Letter Segmentation

Uses a Convolutional Neural Network from TensorFlow (Implemented by Google Vision API)
library to identify the letters in the given input. Using the above information the letters are extracted and cropped to be finally fed into the neural network.

6.1.3 Image Processing

6.1.3.1 Noise Removal

Noise is removed by using a Neural Network that has been trained with noise and noise less Images.

6.1.3.2 Grayscale Conversion

All pictures have 3 channels. Only one color channel is kept and other two are dropped. This converts it into Greyscale.

6.1.3.3 1D Vector Conversion

All Machine Learning libraries provide ways to convert to 1D array. It cab ne easily done.

6.1.4 Machine Learning

Only forward propagation is implemented in the App.

6.1.4.1 Number Detection Module

Uses a Convolutional Neural Network with 32*32*3 input neurons and 10 output neurons. Further uses Regularization with lambda=1 to Make sure overfitting or high variance does not occur.

6.1.4.2 Character Detection Module

Uses a Convolutional Neural Network with 32*32*3 input neurons and 52 output neurons. Further uses Regularization with lambda=1 to Make sure overfitting or high variance does not occur.

6.1.4.3 Special Character Module

Uses a Convolutional Neural Network with 32*32*3 input neurons and 52 output neurons. Further uses Regularization with pooling and dropout layers.

6.1.5 Output

The output Unit is responsible for outputting the proper output. If the chances of being any number is greater than 50% then that number is shown as output.

.

6.2 System Installation Steps

1. Download the android app from the given link in step 2
2. https://www.dropbox.com/s/dbiua1g1yz0zrx7/proOCR.apk?dl=0
3. Go to Security settings on phone and tick “Allow Installation Source” Option
4. Click on the apk and give the required permissions for the app to function.

6.3 System Usage Instructions

6.3.1 Training Instructions

1. A trained model can be created using the python script provided. Install Linux. If Only used for digits then scripts will automatically download the dataset and start training.
2. In the home directory create a folder called “tf_files”. Inside that create a folder called “flower_photos”.
3. In flower Photos create sub folders. The names of the subfolders are class names and the images inside them will be used to train the file.
4. However, the minimum training time is around 2 weeks on a computer with 8 superfast GPU’s.
5. Therefore, it is advised to use Google’s pretrained model.

6.3.2 Running Instructions

1. Click on allow when the app asks for permission

2. Point the camera towards.

3. The detected text will be shown in green . Click convert when detected to get editable text.

7 CONCLUSION

7.1 Project Benefits

Unlike OCRs, ICRs can recognize handwritten text with incredible accuracy because of the neural network based learning algorithms.
Today’s letter sorting machines use OCRs. These machines try to read the pin code of a letter and classify them accordingly. Although this speeds up the process, but still a part of the process is done by hand. Human intervention is

needed when the OCR fails to read the pin code. ICRs will eliminate the need for human intervention as it can have an accuracy of more than 97%

ICRs are also useful for automated forms processing. Many organization use handwritten forms. OCRs are not suitable as they are unable to read handwriting with good amount of accuracy. But ICRs are perfect for the job.

ICR can be used to help people with eye problems too read text that's not clearly visible.

E.g.: ATM pin provided by bank is often difficult to read due to the way they are printed. They often have black stains of ink blocking them. Therefore, people with eye disability are unable to read properly. These people can use ICRs to read the required information.

7.2 Future Scope of Improvement

ICRs read character by character. Therefore, they sometimes cannot read continuous handwriting. In order to overcome this problem Intelligent Word Readers can be built that process one word at a time. This can lead to much better performance in continuous handwriting.

The recognition of ICRs is based on the principle of eye sight i.e. it tries to mimic the way human eyes work. But there are cases when even if the human eye is not able to understand a particular letter in a word, the can human determines it based on the meaning of the sentence. This conscience can be implemented into the computer using Natural Language Processing, Artificial Intelligence and incredibly advanced machine learning.

8 REFERENCES

[1] Priyanka Sharma, Manavjeet Kaur, ―Classification in Pattern Recognition: A Review‖, International Journal of Advanced Research in Computer Science and Software Engineering, Volume 3, Issue 4, April 2013.

[2] Jie Liu, Jigui Sun, Shengsheng Wang, ―Pattern Recognition: An Overview‖, International Journal of Computer Science and Network Security, VOL.6 No.6, June 2006.

[3] A.A Zaidan, B.B Zaidan, Hamid.A.Jalab, Hamdan.O.Alanazi and Rami Alnaqeib, ―Offline Arabic Handwriting Recognition Using Artificial Neural Network‖, Journal of Computer Science and Engineering, Volume 1, Issue 1, May 2010.

[4] V.Kalaichelvi, Ahammed Shamir Ali, ―Application of Neural Network in Character Recognition‖, International Journal of Computer Applications (0975 – 8887) Volume 52– No.12, August 2012.

[5] Divakar Yadav, Sonia Sánchez Cuadrado, Jorge Morato, ―Optical Character Recognition for Hindi Language Using a Neural-network Approach‖, J Inf Process Syst, Vol.9, No.1, March 2013.

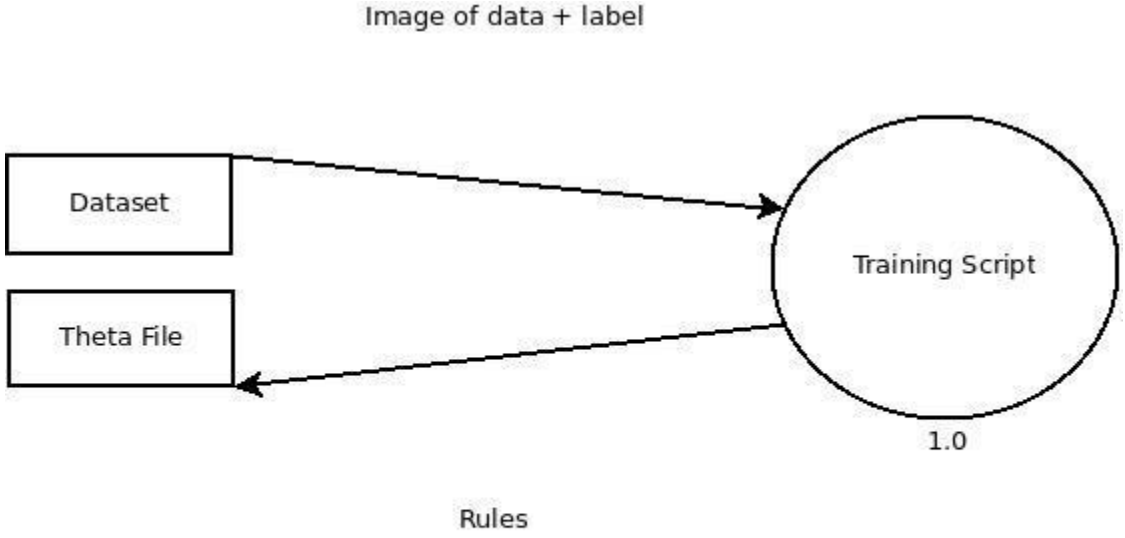
[6] Vidushi Sharma, Sachin Rai, Anurag Dev, ―A Comprehensive Study of Artificial Neural Networks‖, International Journal of Advanced Research in Computer Science and Software Engineering, Volume 2, Issue 10, October 2012.

[7] Jyoti Singh, Pritee Gupta, ―Advance Applications of Artificial Intelligence and Neural Networks: A Review‖, AKGEC JOURNAL OF TECHNOLOGY, vol.1, no.2.

[8] Chirag I Patel, Ripal Patel, Palak Patel, ―Handwritten Character Recognition using Neural Network‖, International Journal of Scientific & Engineering Research Volume 2, Issue 5, May-2011.

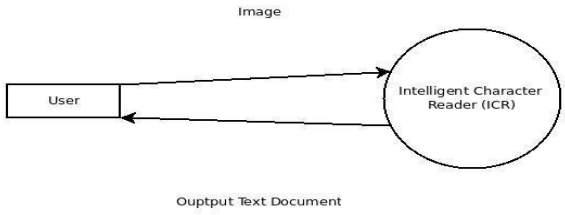
[9] Guobin Ou, Yi Lu Murphey, ―Multi-class pattern classification using neural networks‖, A Journal of the Pattern Recognition Society‖, Vol 40, 2007, pg 4-18.
[10] Guobin Ou, Yi Lu Murphey, ―Multi-class pattern classification using neural networks‖, Journal of Pattern Recognition, Volume 40, Issue 1, January, 2007.

APPENDIX-A

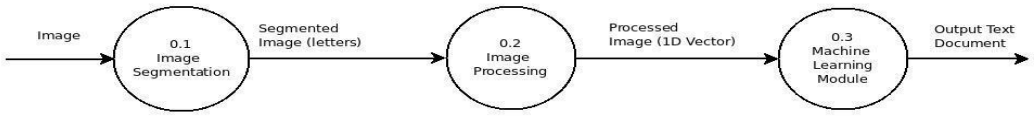


(a) LEVEL 0 DFD
(Context Diagram for Training Script)

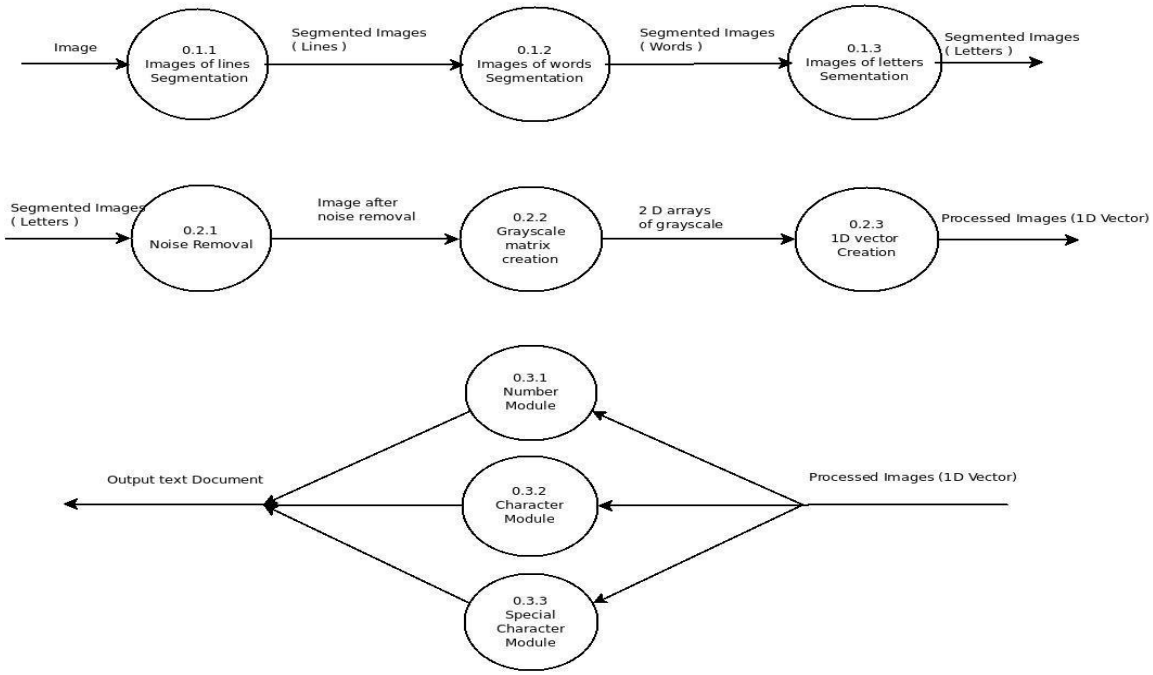
Figure 1: DFD for training



LEVEL 0 DFD (Context Diagram)



LEVEL 1 DFD



LEVEL 2 DFD

Figure 2: Data flow diagrams

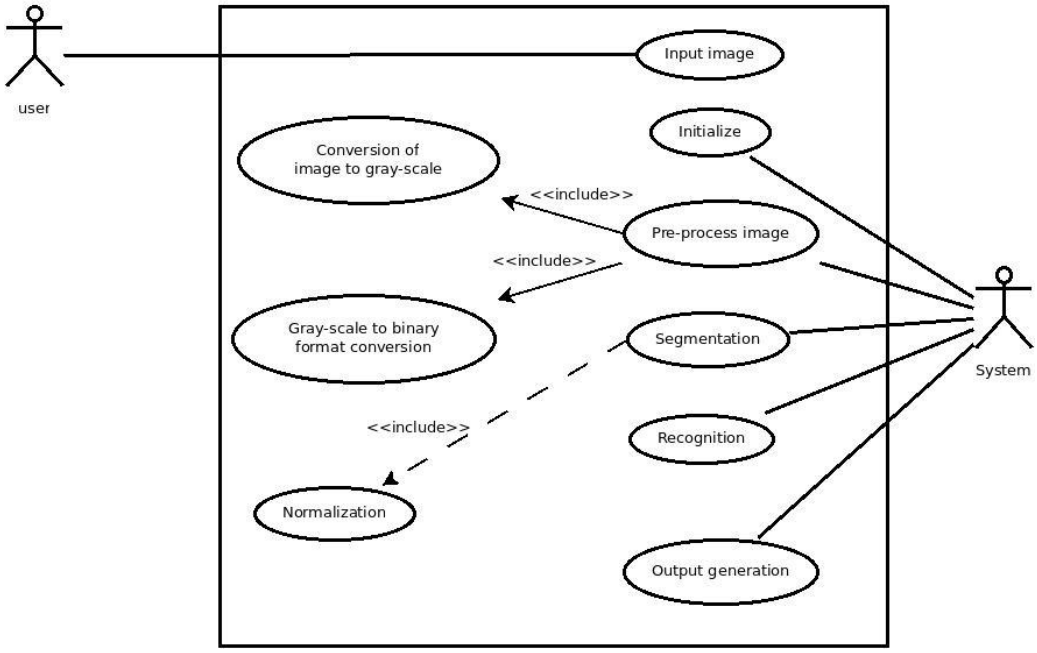


Figure 3: Use case diagram

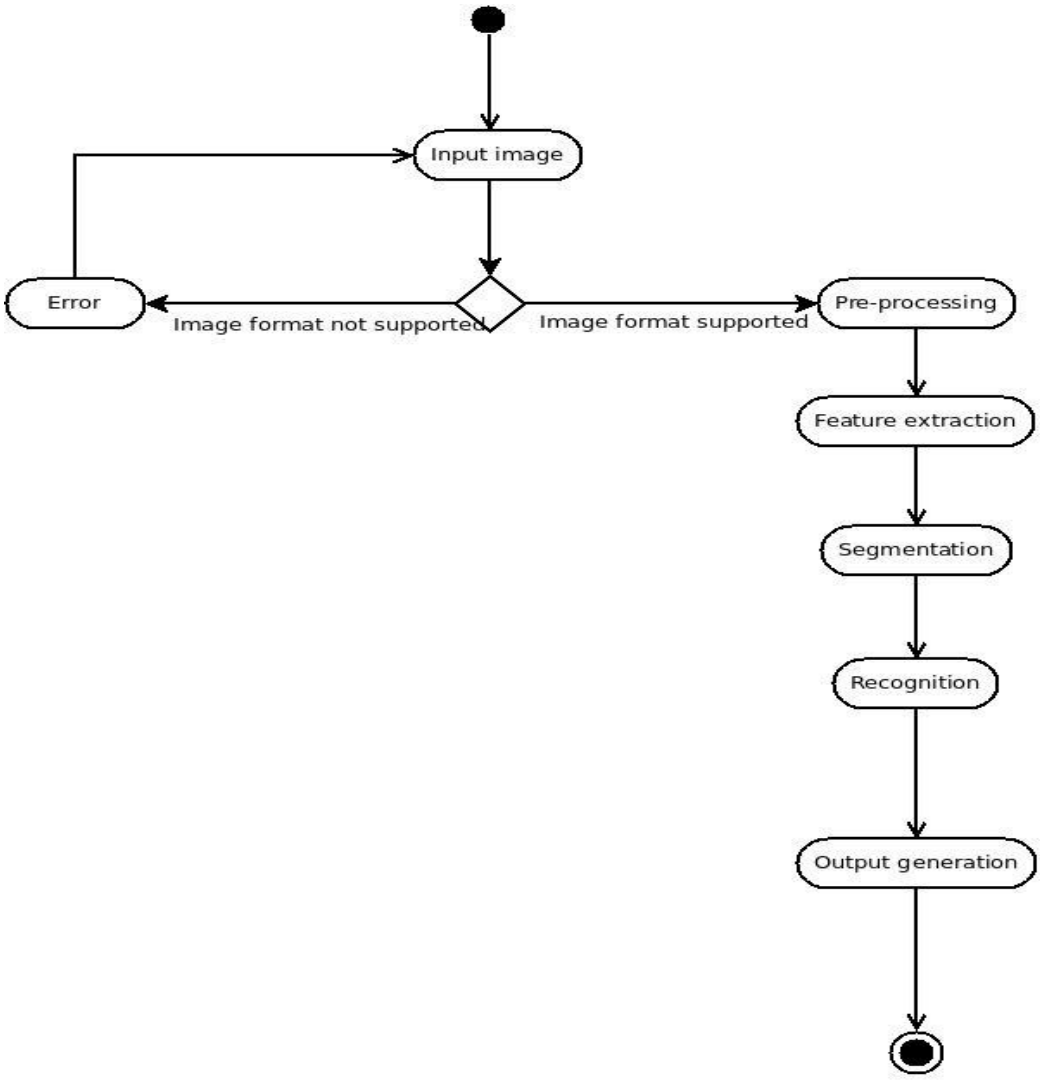


Figure 4: Activity Diagram

APPENDIX-B

This part contains only the main parts of the code and not the full code.

The machine Learning Code:

```
private void createCameraSource(boolean autoFocus, boolean useFlash) {
    Context context = getContext();

    // TODO: Create the TextRecognizer
    TextRecognizer textRecognizer = new
TextRecognizer.Builder(context).build();
    // TODO: Set the TextRecognizer's Processor.
    textRecognizer.setProcessor(new
OcrDetectorProcessor(mGraphicOverlay,button1,fragm));
    // TODO: Check if the TextRecognizer is operational.
    if(!textRecognizer.isOperational()) {
        Log.w(TAG,"Detector Dependencies are still not available");
        // Check for low storage.
        IntentFilter intentFilter = new
IntentFilter(Intent.ACTION_DEVICE_STORAGE_LOW);
        boolean boo_storage_check =
getContext().registerReceiver(null,intentFilter) !=null;

        if(boo_storage_check){

Toast.makeText(getContext(),R.string.low_storage_error,Toast.LENGTH_LONG).s
how();
            Log.w(TAG,getString(R.string.low_storage_error));
        }
    }
    // TODO: Create the mCameraSource using the TextRecognizer.
    mCameraSource = new
CameraSource.Builder(getContext(),textRecognizer).setFacing(CameraSource.CA
MERA_FACING_BACK).setRequestedPreviewSize(1280,1024).setRequestedFps(15.0f)
.setFlashMode(useFlash ? Camera.Parameters.FLASH_MODE_TORCH :
null).setFocusMode(autoFocus ?
Camera.Parameters.FOCUS_MODE_CONTINUOUS_PICTURE : null).build();


}
```

The Text detector :

```
public void receiveDetections(Detector.Detections<TextBlock> detections) {
    mGraphicOverlay.clear();
    items = detections.getDetectedItems();
    for(int i=0;i<items.size();++i){
        TextBlock item = items.valueAt(i);
        if(item!=null && item.getValue()!=null){
            Log.d("Processor","Text Detected! "+ item.getValue());
        }
        OcrGraphic graphic = new OcrGraphic(mGraphicOverlay,item);
        mGraphicOverlay.add(graphic);
    }

}


public void onClick(View v) {
    String mytext = "";
    if(v.getId()==R.id.button){
        if(items!=null){
            for(int i=0;i<items.size();++i){
                TextBlock item = items.valueAt(i);
                if(item!=null && item.getValue()!=null){
                    mytext=mytext+item.getValue();
                }
            }
            Log.v("yoyo",mytext);
            Text1 te = new Text1();
            Bundle arguments = new Bundle();
            arguments.putString("Dipayan",mytext);
            te.setArguments(arguments);
            FragmentTransaction ft = context.beginTransaction();
            ft.replace(R.id.dispfrag, te);
            ft.commit();

        }
        else{
            Log.v("Button1","I am getting no string you idiot");
        }
    }
}
```

Similar Python Code which was used to train the TextRecognizer class in Google Cloud App(for the full code please see Buga.py)

```
# The first cov layer. Each filter is of 5*5*1 . There are 32 filters in the first layer.
```

```
x = tf.placeholder(tf.float32,shape=[None,784])
y_ = tf.placeholder(tf.float32, shape=[None, len(d)])
W_conv1 = weight_variable([5,5,1,32])
b_conv1 = bias_variable([32])
x_image = tf.reshape(x,[-1,28,28,1])
# We then convolve x_image with the weight tensor, add the bias, apply the ReLU function
# and finally max pool. The max_pool_2x2 method will reduce the image size to 14x14
h_conv1 = tf.nn.relu(conv2d(x_image, W_conv1)+b_conv1)
h_pool1 = max_pool_2x2(h_conv1)
# The Second Convolutional Layer.
# Has 64 filters
W_conv2 = weight_variable([5,5,32,64])
b_conv2 = bias_variable([64])
h_conv2 = tf.nn.relu(conv2d(h_pool1,W_conv2)+b_conv2)
h_pool2 = max_pool_2x2(h_conv2)
# Densly connected layer
W_fc1 = weight_variable([7*7*64,1024])
b_fc1 = bias_variable([1024])
h_pool2_flat = tf.reshape(h_pool2, [-1,7*7*64])
h_fc1 = tf.nn.relu(tf.matmul(h_pool2_flat,W_fc1)+b_fc1)
#Dropout
keep_prob = tf.placeholder(tf.float32)
h_fc1_drop = tf.nn.dropout(h_fc1, keep_prob)
# Readout Layer
W_fc2 = weight_variable([1024,len(d)])
b_fc2 = bias_variable([len(d)])
y_conv = tf.matmul(h_fc1_drop , W_fc2)+ b_fc2
cross_entropy = tf.reduce_mean(tf.nn.softmax_cross_entropy_with_logits(labels=y_, logits=y_conv))
train_step = tf.train.AdamOptimizer(1e-4).minimize(cross_entropy)
correct_prediction = tf.equal(tf.argmax(y_conv,1), tf.argmax(y_,1))
accuracy = tf.reduce_mean(tf.cast(correct_prediction, tf.float32))
```

```
sess = tf.Session()

sess.run(tf.global_variables_initializer())

'''

amount = len(training_image_data)

times = 1000

for i in range(times):

   input_data = get_next_batch_data(amount,times,i,training_image_data)

   input_index = get_next_batch_class(amount,times,i,index_training)

   train_step.run(session=sess,feed_dict={x: input_data, y_: input_index, keep_prob: 0.5})

   if i%100:

      input_data_testing = get_next_batch_data(amount,times,i,testing_image_data)

      input_index_testing = get_next_batch_class(amount,times,i,index_testing)

      print("test accuracy %g" % accuracy.eval(session=sess,feed_dict={x: input_data_testing, y_: input_index_testing, keep_prob: 1.0}))

'''

train_step.run(session=sess,feed_dict={x: training_image_data, y_: index_training ,keep_prob: 0.5})

print("test accuracy %g"%accuracy.eval(session=sess,feed_dict={x: testing_image_data, y_: index_testing, keep_prob: 1.0}))
```

Output Code :

```
@Override
public View onCreateView(LayoutInflater inflater, ViewGroup container,
                         Bundle savedInstanceState) {
    // Inflate the layout for this fragment
    View view = inflater.inflate(R.layout.fragment_text, container, false);
    textView = (EditText) view.findViewById(R.id.DispOCR);
    mytext = arguments.getString("Dipayan");
    textView.setText(mytext);

    button2 = (Button) view.findViewById(R.id.button2);
     //       tts.speak(mytext, TextToSpeech.QUEUE_ADD, null, "DEFAULT");
    button2.setOnClickListener(this);
    return view;
}
```

The working app

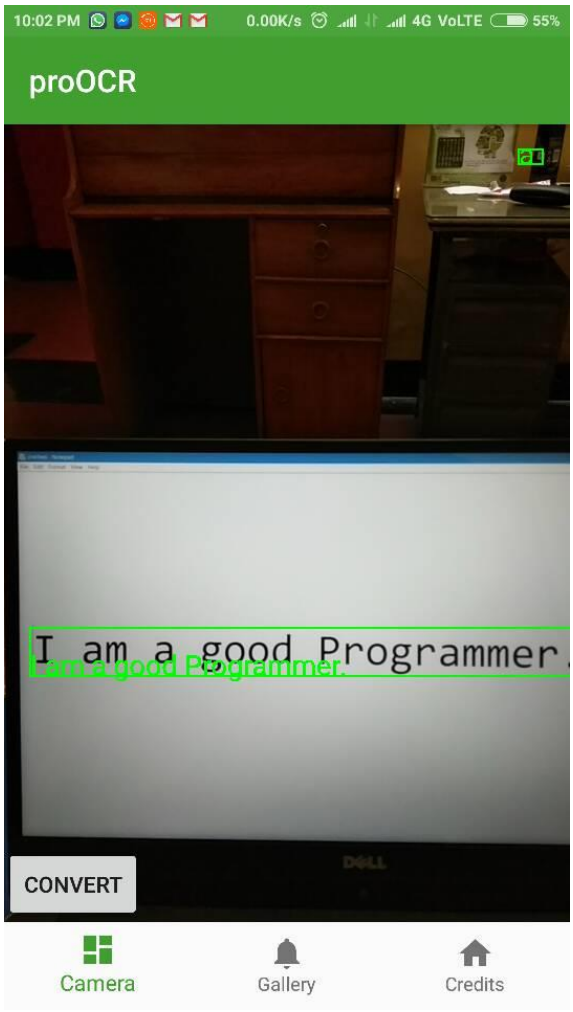


Figure 5

Figure 6

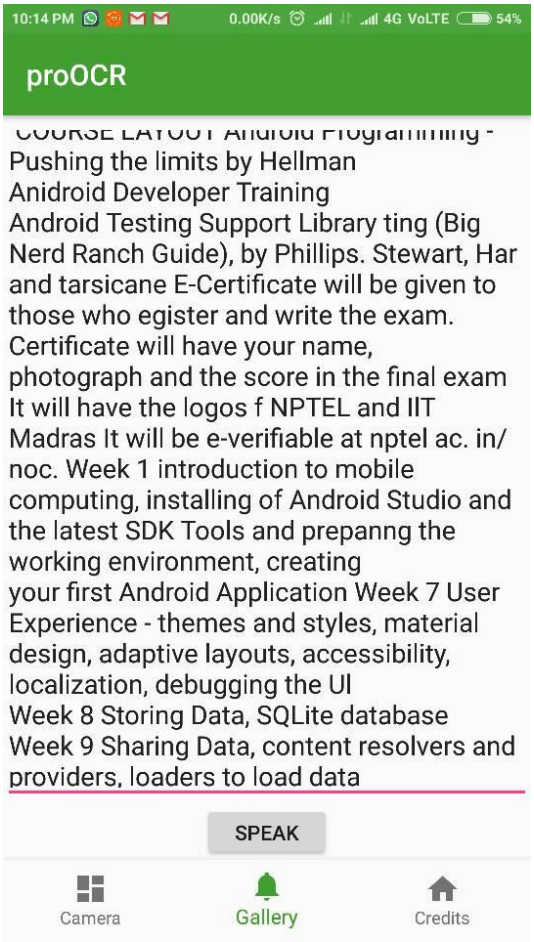


Figure 7

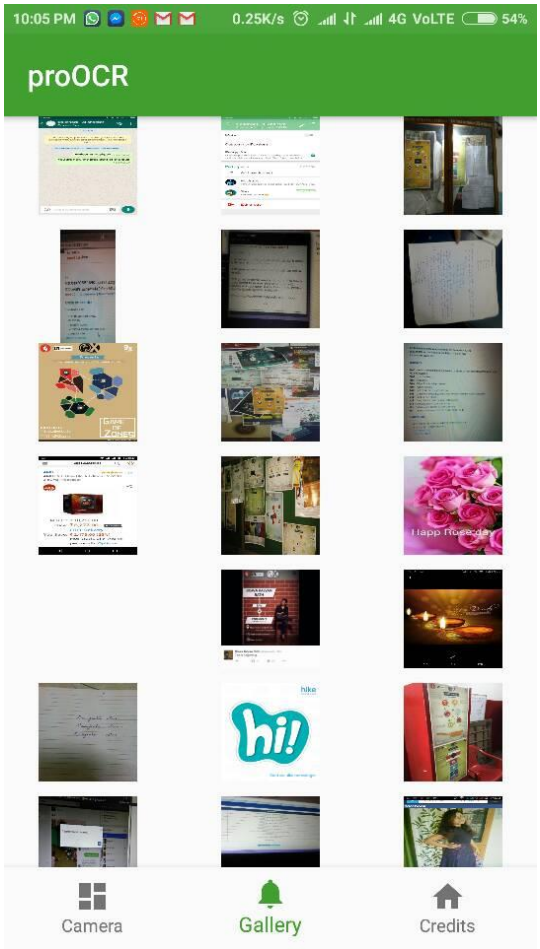


Figure 8