INDUSTRIAL TRAINING REPORT

ON

OPTICAL CHARACTER RECOGNITION(OCR)

Submitted in partial fulfilment of the
Requirements for the award of the degree

Of

Bachelor of Technology

In

Computer Science and Engineering

by

Taranjyot Singh(09413202715/CSE 2)

Department of Computer Science

Guru Tegh Bahadur Institute Of Technology

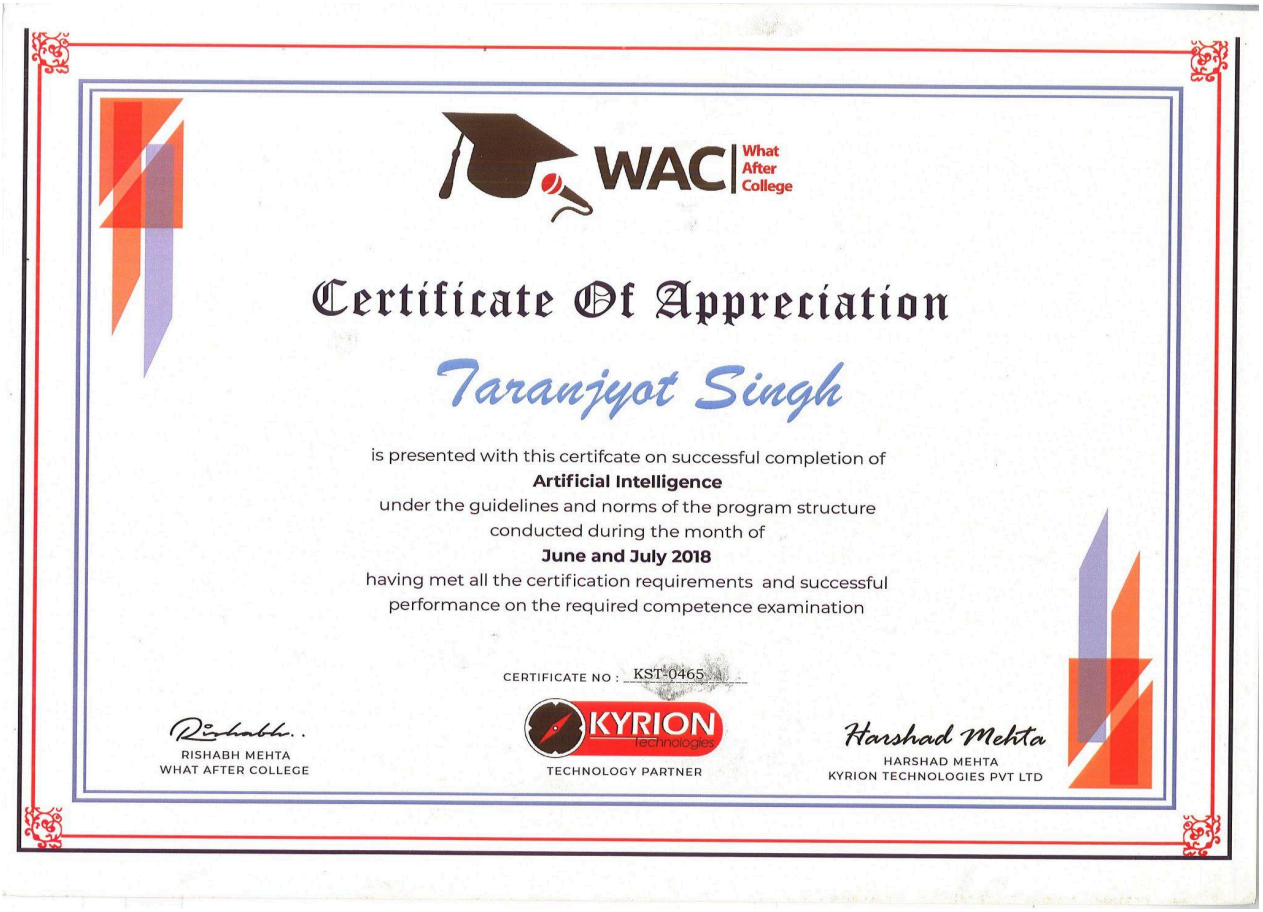
Rajouri , New Delhi.

Under

Guru Gobind Singh Indraprastha University

Dwarka , New Delhi.

CERTIFICATE



ACKNOWLEDGEMENT

I have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals and organizations. I would like to extend my sincere thanks to all of them.

I am highly indebted to WAC, Connaught Place for their guidance and constant supervision as well as for providing necessary information regarding the project and also for their support in completing the project.

I consider myself lucky enough to get such a good project. This project would add as an asset to my academic profile.

I would like to express my thankfulness to my project guide, **Mr. Vishwajeet Singh Rana** for his constant motivation and valuable help through the project work, and I express my gratitude to **Mr. Rishabh Mehta**, for his constant supervision, guidance and co-operation throughout the project.

I would like to express my great gratitude towards our supervisor **Mrs. Geetika Bhatia** who has given me support and suggestions.

Finally I would like to thank my friends and parents for their co-operation to complete this project.

Date: October 15, 2018

Taranjyot Singh

(094/CSE2/2015)

taranjyotsingh357@gmail.com

ABSTRACT

In this document I am presenting the requirements specification of the project Optical Character Recognition.

OCR stands for Optical Character Recognition. OCR technology is a software that scans documents containing texts and converts them into documents that can be edited. However, for the scanning to take place, the text should be clear and at times, handwritten text may not be recognized by the software. This report presents the OCR handwriting recognition in which we can actually scan a piece of paper with a structured format, such as writing one letter per box on highly structured application forms, and the scanner will be able to convert the text into a digital format.

The various stages of OCR are: upload a scan image from the computer, segmentation process in which we extract the text zone from the image, recognition of the text and the last which is the post processing process in which the output of the previous stage goes through the error detection and error phase.

TABLE OF CONTENTS

LIST OF FIGURES

Chapter 1
INTRODUCTION

In the running world, there is growing demand for the software systems to recognize characters in computer systems when information is scanned through paper documents as we know that we have a number of newspapers and books which are in printed format related to different subjects. These days there is a huge demand in "storing the information available in these paper documents into a computer storage disk and then later reusing this information by searching process". One simple way to store information in these paper documents into a computer system is to first scan the documents and then store them as IMAGES. But to reuse this information it is very difficult to read the individual contents and search the contents from these documents line-by-line and word-by-word. The reason for this difficulty is that the font characteristics of the characters in paper documents are different from the font of the characters in the computer system.

As a result, the computer is unable to recognize the characters while reading them. This concept of storing the contents of paper documents in a computer storage place and then reading and searching the content is called DOCUMENT PROCESSING. Sometimes in this document processing we need to process the information that is related to languages other than the English in the world. For this document processing we need a software system called **CHARACTER RECOGNITION SYSTEM**. This process is also called **DOCUMENT IMAGE ANALYSIS (DIA)**. Thus our need is to develop a character recognition software system to perform Document Image Analysis which transforms documents in paper format to electronic format. For this process there are various techniques in the world. Among all those techniques we have 2 chosen Optical Character Recognition as the main fundamental technique to recognize characters. The conversion of paper documents into electronic format is an on-going task in many of the organizations particularly in the Research and Development (R&D) area, in large business enterprises, in government institutions, and so on. From our problem statement we can introduce the necessity of Optical Character Recognition in mobile electronic devices such as cell phones, digital cameras to acquire images and recognize them as a part of face recognition and validation. To effectively use Optical Character Recognition for character recognition in-order to perform Document Image Analysis (DIA), we are using the information in Grid format. . This system is thus effective and useful in Virtual Digital Library's design and construction.

1.1 PURPOSE

The main purpose of the Optical Character Recognition (OCR) system based on a grid infrastructure is to perform Document Image Analysis, document processing of electronic document formats converted from paper formats more effectively and efficiently. This improves the accuracy of recognizing the characters during document processing compared to various existing available character recognition methods. Here OCR technique derives the meaning of the characters, their font properties from their bit-mapped images.

- The primary objective is to speed up the process of character recognition in document processing. As a result the system can process a huge number of documents with-in less time and hence saves the time.
- Since our character recognition is based on a grid infrastructure, it aims to recognize multiple heterogeneous characters that belong to different universal languages with different font properties and alignments.

1.2 PROJECT SCOPE

The scope of our product Optical Character Recognition on a grid infrastructure is to provide an efficient and enhanced software tool for the users to perform Document Image Analysis, document processing by reading and recognizing the characters in research, academic, governmental and business organizations that are having large pools of documented, scanned images. Irrespective of the size of documents and the type of characters in documents, the product is recognizing them, searching them and processing them faster according to the needs of the environment.

1.3 EXISTING SYSTEM

In the running world there is a growing demand for the users to convert the printed documents into electronic documents for maintaining the security of their data. Hence the basic OCR system was invented to convert the data available on papers into computer process-able documents, So that the documents can be editable and reusable. The existing system/the previous system of OCR on a grid infrastructure is just OCR without grid functionality. That is the existing system deals with the homogeneous character recognition or character recognition of single languages.

1.4 DRAWBACK OF EXISTING SYSTEM

The drawback in the early OCR systems is that they only have the capability to convert and recognize only the documents of English or a specific language only. That is, the older OCR system is unilingual.

1.5 PROPOSED SYSTEM

Our proposed system is OCR on a grid infrastructure which is a character recognition system that supports recognition of the characters of multiple languages. This feature is what we call grid infrastructure which eliminates the problem of heterogeneous character recognition and supports multiple functionalities to be performed on the document. The multiple functionalities include editing and searching too whereas the existing system supports only editing of the document. In this context, Grid infrastructure means the infrastructure that supports a group of specific sets of languages. Thus OCR on a grid infrastructure is multilingual.

1.6 BENEFIT OF PROPOSED SYSTEM

The benefit of the proposed system that overcomes the drawback of the existing system is that it supports multiple functionalities such as editing and searching. It also adds benefit by providing heterogeneous character recognition.

1.7 ARCHITECTURE OF THE PROPOSED SYSTEM

The Architecture of the optical character recognition system on a grid infrastructure consists of the three main components. They are:

- Scanner
- OCR Hardware or Software
- Output Interface

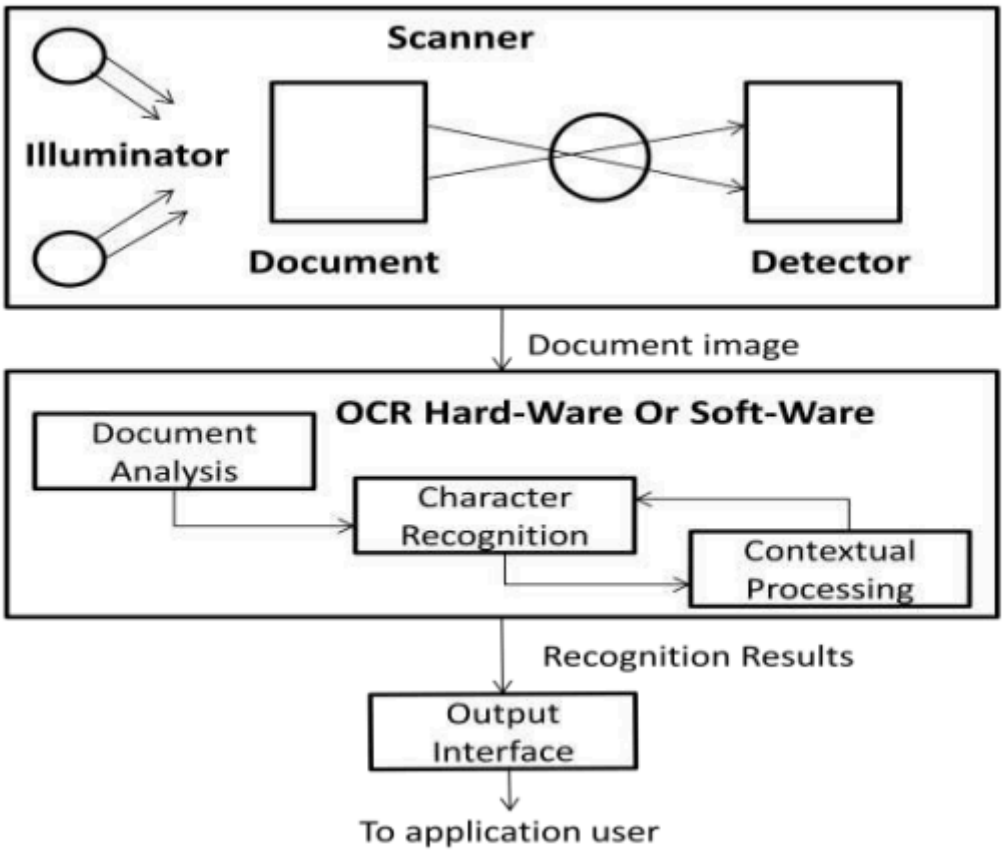


Fig1 : OCR Architecture

Chapter 2

REQUIREMENT ANALYSIS (SRS)

2.1 Hardware Requirements

Hardware: Hardware is the physical part of the computer system like mouse, keyboard, monitor etc.

Hardware Requirements	
Processor	Pentium IV or higher
Hard Disk	500MB or higher
RAM	minimum 512MB
Screen resolution	1280 x 800
Server	
Keyboard and mouse	

2.2 Software Requirements

Software: Software is a set of applications which is used to run the operating system.

Types of Software are:

- System Software
- Application Software

Software Requirements	
Operating System	Windows XP/ Windows 7/8
User Interface	Swings
Programming Language	Python
Database	Access/MySQL

2.3 Functional Requirements

- Maintenance of user’s expenses which are hard to maintain in today's busy life.
- Analysis of expenses.
- Categories according to user’s preferences.
- Keeping track of the money the user has borrowed from or lent to a friend/contact.
- Backup/Restore the data to Google Drive.

2.4 Non-Functional Requirements

- Product must be reliable.
- Simple and easy to use.
- It must be maintained after users start using the product.
- It must be secure.
- It must be effective and quality should be good.
- It must support various versions of android.

Chapter 3
SYSTEM DESIGN (SDS)

3.1 CLIENT SERVER ARCHITECTURE

In the deployment diagram of our OCR system, the server role is played by an admin called Librarian. There can be N number of clients who can access the digital library data content at a time. The clients here may be either the students or the faculty or both.

The actions performed by the Administrator are document processing, searching and editing whereas the actions performed by the end-user are only document searching and editing.

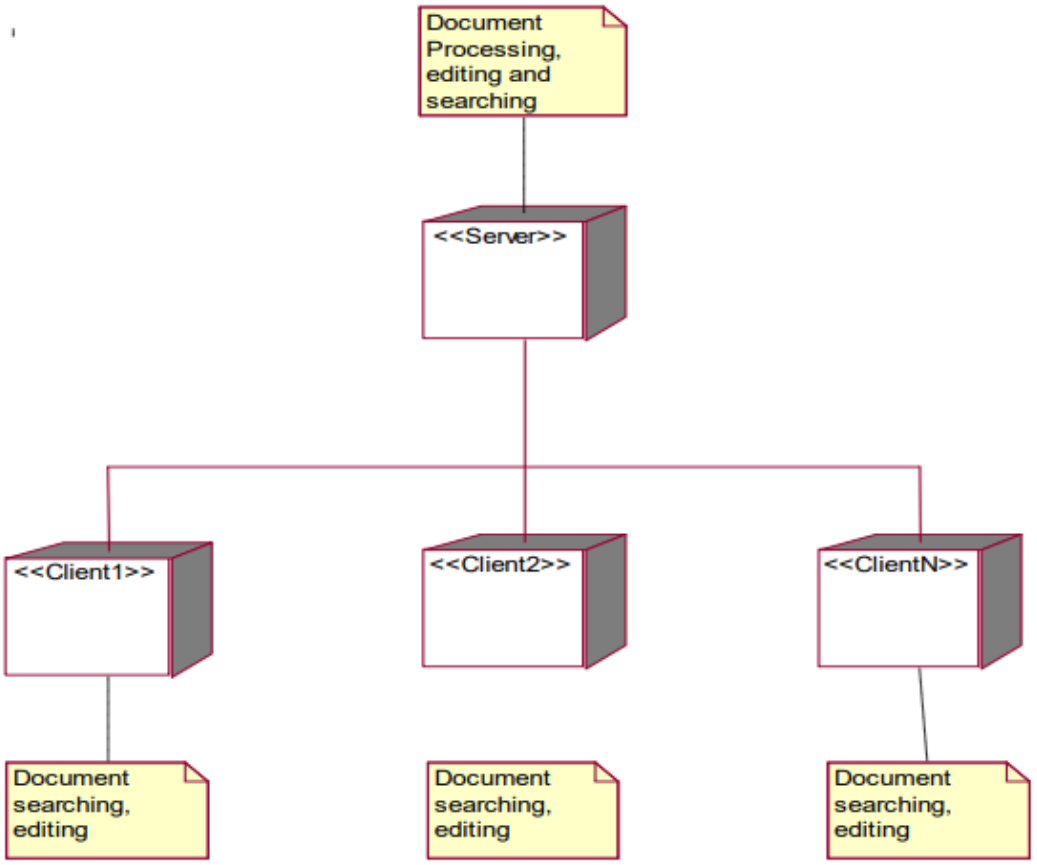


Fig2 : Client-Server Architecture

3.2 DATA FLOW DIAGRAM

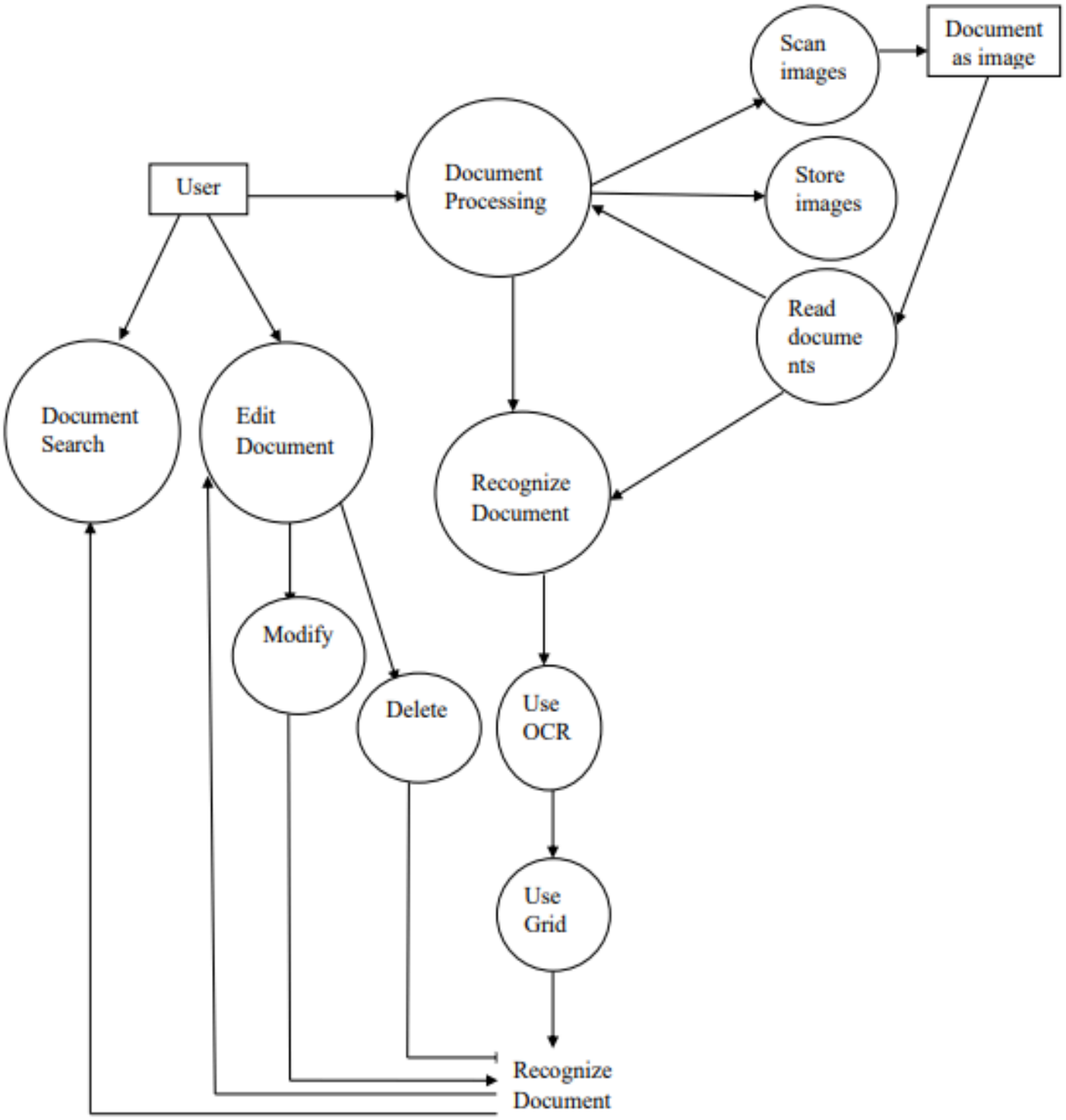
The DFD is also called a bubble chart. A data-flow diagram (DFD) is a graphical representation of the "flow" of data through an information system. DFD's can also be used for the visualization of data processing. The flow of data in our system can be described in the form of data flow diagram as follows:-

1) Firstly, if the user is administrator he can initialize the following actions:
   - Document processing
   - Document search
   - Document editing

All the above actions come under 2 cases. They are described as follows:

   a) If the printed document is a new document that is not yet read into the system, then the document processing phase reads the scanned document as an image only and then produces the document image stored in computer memory as a result. Now the document processing phase has the document at its hand and can read the document at any point of time. Later the document processing phase proceeds with recognizing the document using OCR methodology and the grid infrastructures. Thus it produces the documents with the recognized characters as final output which can be later searched and edited by the end-user or administrator.
   b) If the printed document is already scanned in and is held in system memory, then the document processing phase proceeds with document recognition using OCR methodology and grid infrastructure. And thus it finally produces the document with recognized documents as output.

2) If the user using the OCR system is the end-user, then he can perform the following actions:-
   - Document searching : The documents which are recognized can be searched by the user whenever required by requesting from the system database.
   - Document editing : The recognized documents can be edited by adding the specific content to the document, deleting specific content from the document and modifying the document.



Data Flow Diagram

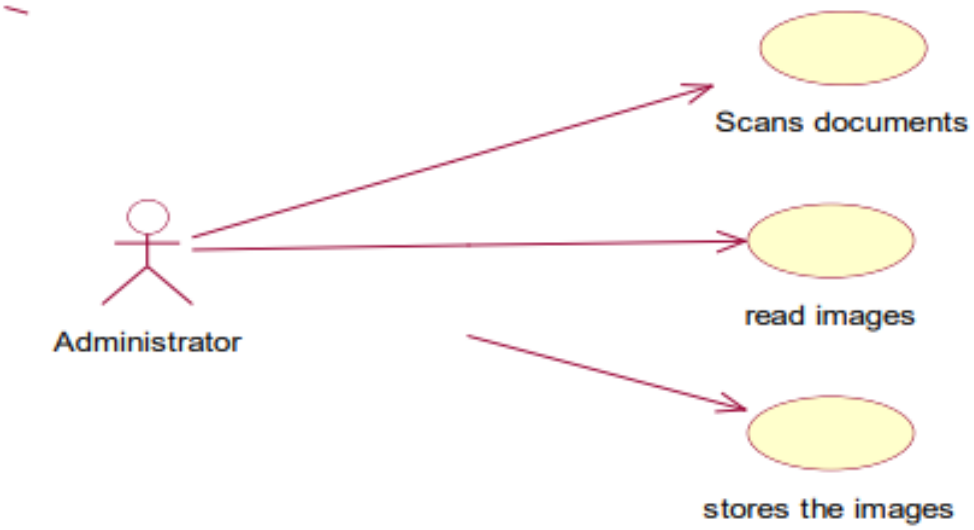
3.3 USE-CASE DIAGRAMS

Our software system can be used to support the library environment to create a Digital Library where several paper documents are converted into electronic-form for accessing by the users. For this purpose the printed documents must be recognized before they are converted into electronic-form. The resulting electronic-documents are accessed by the users like 25 faculty and students for reading and editing. Now according to this information, the following are the different actors involved in implementing our OCR system:-

- If we consider a virtual digital library, the Administrator can be the Librarian and the End-users can be Students or/and Faculty.
- The following are the list of use diagrams that altogether form the complete or the overall use-case diagram. They are listed below:-
    - Use-case diagram for document processing
    - Use-case diagram for neural network training
    - Use-case diagram for document recognition
    - Use-case diagram for document editing
    - Use-case diagram for document searching

In each of the use-case diagrams below we clearly explained about that particular use case functionality. In this we provided a description about the

- Use-case name
- Details about the use-case
- Actors using this use-case
- The flow of events carried out by the use-case
- The conditions that occur in this use-case



Use-Case Diagram for Document Processing

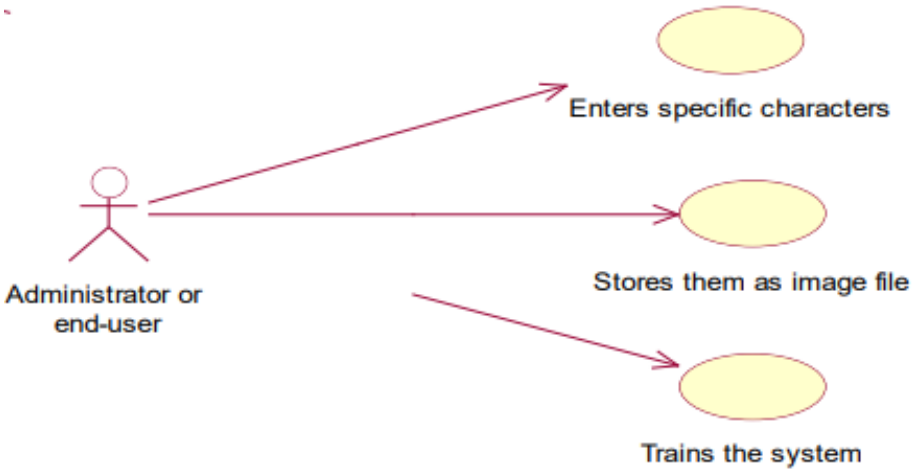
Use Case Name :
Document processing

Description :
The administrator is the only person who participates in the document processing. Here he scans the documents. The scanned documents are read as images. Finally the read images are stored in the system memory.

Actors :
Primary Actor : Administrator
Secondary Actor : User

Flow of Events :

1. The Administrator scans the document which he wants to edit.
2. The scanned documents are read as images.
3. Finally the images that are read are stored in system memory for future reference.



Use-Case Diagram for Neural Network Training

Use case Name :
Neural Network Training

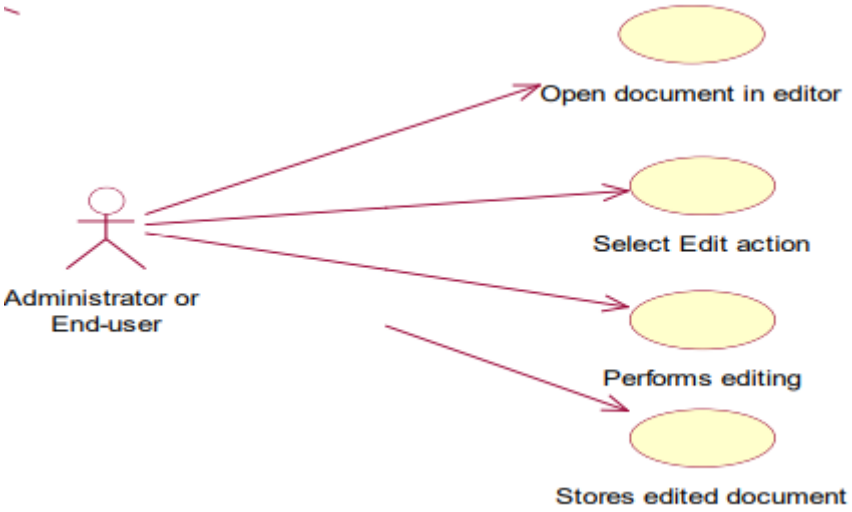
Description :
The Administrator or End-user enters the specific characters required for training. User stores them as image files and trains the system.

Actors :
Primary Actor : Administrator or End-user
Secondary Actor : User

Flow of Events :

1. The user enters the specific characters in order to train the system.
2. After entering it is stored as an image file.
3. Finally trains the system according to the system.

Pre-Condition :
The font in the scanned document should be identified.



Use-Case Diagram for Document Editing

Use case Name :
Document editing

Description :
Both Administrator and End-user can perform the document editing. The user opens the document in the editor and selects the edit action i.e., edit, modify, delete etc. After selecting the edit action editing operation is performed and finally stores the document that had been edited.

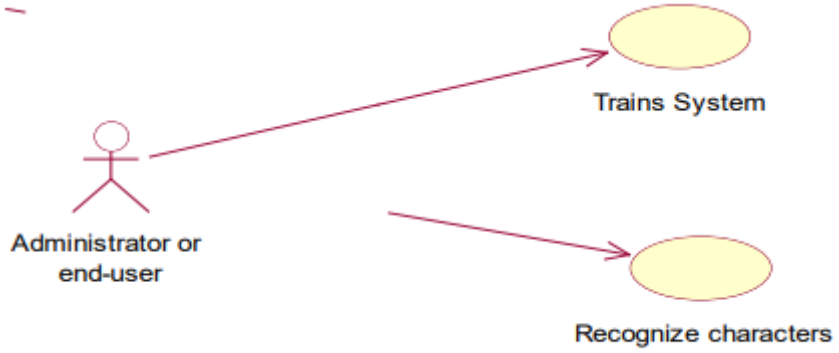
Actors :
Primary Actor : Administrator or End-user
Secondary Actor : User

Flow of Events :

1. The Administrator or End-user opens the document which he wants to edit.
2. He selects the edit action. The action consists of editing the document, modifying the document, deleting the document etc.
3. After selecting the edit action the editing operation is performed.
4. Finally the edited document is stored in the system memory.

Pre-Condition :
The input to be taken for editing should be an image of the document that is converted in to word or text file. That is the input file must be either .doc file or .txt file only.

Post-Condition :
Finally after editing the document there are specific target formats defined by the user. The document should be saved in that format only. That will be the output of the editor. That is, as per our design the final document after editing must be saved in .doc file or .txt file only.



Use-Case Diagram for Document Recognition

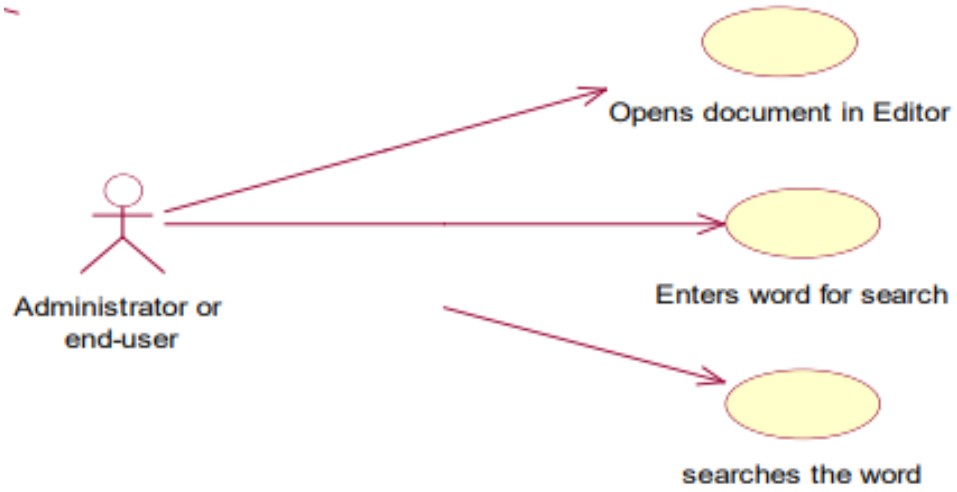
Use case Name :
Document Recognition

Description :
The Administrator or End-user trains the system according to the given symbols or alphabets. Then the characters are recognized after the system is trained.

Actors :
Primary Actor : Administrator or End-user
Secondary Actor : User

Flow of Events :
1. The user trains the system to recognize the characters.
2. After the system is trained the characters are recognized.

Pre-Condition :
Before trying to recognize the characters, the system should be trained first with the font characteristics and the font size.



Use-Case Diagram for Document Searching

Use case Name :
Document Searching

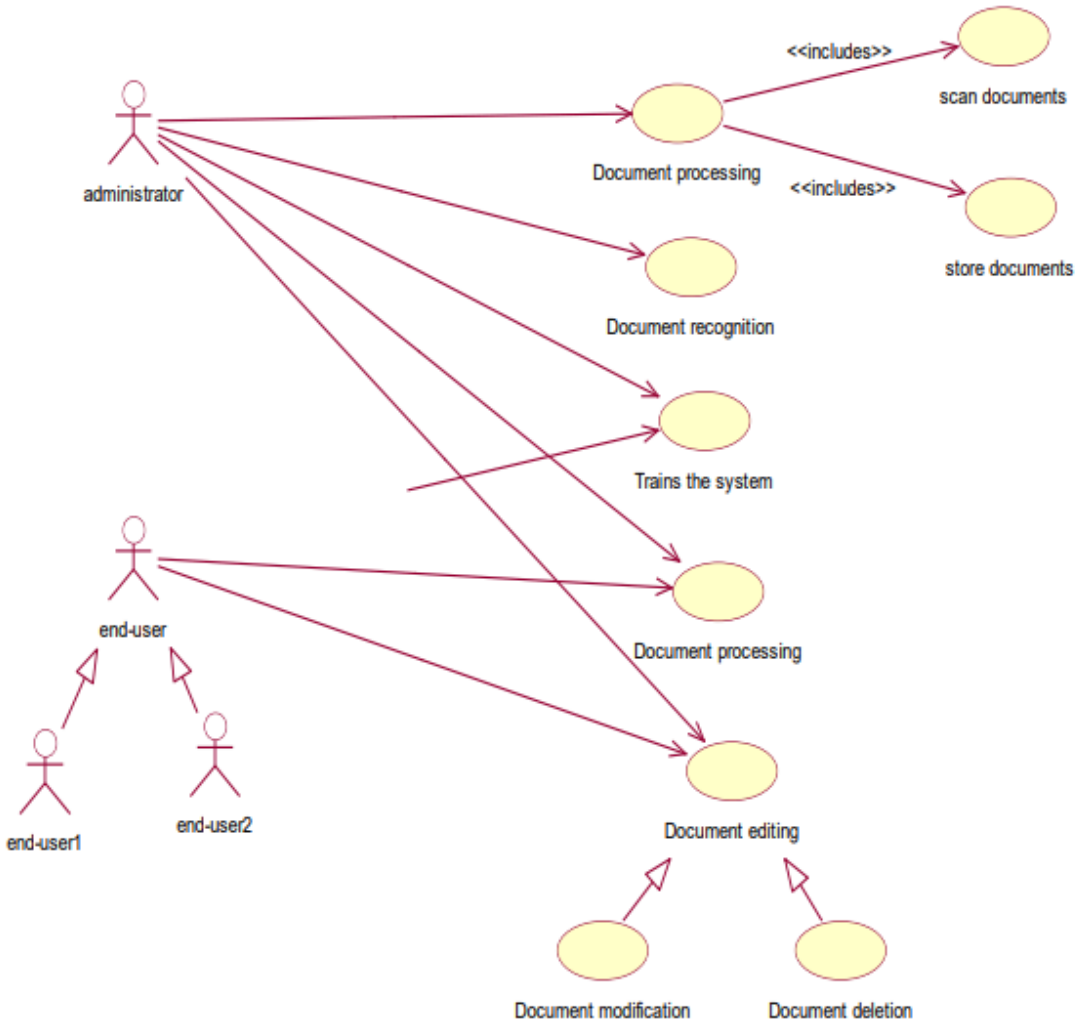
Description :
The Administrator or End-user opens the document in the editor. He enters the word which he is looking for in that document. Then he searches for the word.

Actors :
Primary Actor : Administrator or End-user
Secondary Actor : User

Flow of Events :

1. The user opens the document for searching a word he requires.
2. After opening the document he enters the word for search.
3. Finally searches the word in that document.

Pre And Post Conditions :
No pre-condition and post-condition.



Overall Use-Case Diagram

Chapter 4
TECHNOLOGY/CONCEPT USED

4.1 LANGUAGE USED : PYTHON

Python is an interpreted high-level programming language for general-purpose programming. Python has a design philosophy that emphasizes code readability, notably using significant whitespace. It provides constructs that enable clear programming on both small and large scales.

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object oriented, imperative, functional and procedural and has a large and comprehensive standard library.

Python interpreters are available for many operating systems. Python's large standard library, commonly cited as one of its greatest strengths, provides tools suited to many tasks. For Internet-facing applications, many standard formats and protocols such as MIME and HTTP are supported. It includes modules for creating graphical user interfaces, connecting to relational databases, generating pseudorandom numbers, arithmetic with arbitrary precision decimals, manipulating regular expressions, and unit testing.

As of March 2018, the Python Package Index (PyPI), the official repository for third-party Python software, contains over 130,000 packages with a wide range of functionality, including:

- Graphical user interfaces
- Web frameworks
- Multimedia
- Databases
- Networking
- Test frameworks
- Automation
- Web scraping
- Documentation
- System administration
- Scientific computing
- Text processing
- Image processing

Python's design and philosophy have influenced many other programming languages:

- Boo uses indentation, a similar syntax, and a similar object model.
- Cobra uses indentation and a similar syntax, and its "Acknowledgements" document lists Python first among languages that influenced it.] However, Cobra directly supports design-by-contract, unit tests, and optional static typing.
- Kotlin is a functional programming language with an interactive shell similar to Python. However, Kotlin is strongly typed with access to standard Java libraries.
- Ruby's creator, Yukihiro Matsumoto, has said: "I wanted a scripting language that was more powerful than Perl, and more object-oriented than Python. That's why I decided to design my own language."

Why Python for Artificial Intelligence & Machine Learning?

Whether a startup or an MNC, Python provides a huge list of benefits to all. The usage of Python is such that it cannot be limited to only one activity. Its growing popularity has

allowed it to enter into some of the most popular and complex processes like Artificial Intelligence (AI), Machine Learning (ML), natural language processing, data science etc.

1. Less Code : AI involves algorithms - a LOT of them. Python provides ease of testing - one of the best among competitors. Python helps in easy writing and execution of codes. Python can implement the same logic with as much as 1/5th code as compared to other OOPs languages. Thanks to its interpreted approach which enables check as you code methodology.

2. Prebuilt Libraries : Python has a lot of libraries for every need of your AI project. Few names include Numpy for scientific computation, Scipy for advanced computing and Pybrain for machine learning. AIMA - Python implementation of algorithms from Russell and Norvig's 'Artificial Intelligence: A Modern Approach' is one of the best library available for Artificial Intelligence till today. Such a dedicated library saves developer's time spent on coding base level items.

3. Support : Python is a completely open source with a great community. There are a host of resources available which can get any developer up to speed in no time. Not to forget, there is a huge community of active coders willing to help programmers in every stage of the development cycle.

4. Platform Agnostic : Python provides the flexibility to provide an API from an existing language which indeed provides extreme flexibility. It is also platform independent. With just a few changes in codes, you can get your app up and running in a new OS. This saves developers time in testing on different platforms and migrating code.

5. Flexibility : Flexibility is one of the core advantages of Python. With the option to choose between OOPs approach and scripting, Python is suitable for every purpose. It works as a perfect backend and it is also suitable for linking different data structures together. The option to check a majority of code in the IDE itself is also a big plus for developers who are struggling between different algorithms.

6. Popularity : Python is winning the heart of millennials. Its ease of learning is attracting millennials to learn this language. Though AI Projects need a highly experienced programmer yet Python can smoothen the learning curve. It is practically easier to look for Python developers than to hunt for LISP or Prolog programmers, particularly in some nations. Its extended libraries and active community with an ever developing and improving code have led it to be one of the hottest languages today.

4.2 PYCHARM

PyCharm is an integrated development environment (IDE) used in computer programming, specifically for the Python language. It is developed by the Czech company JetBrains. It provides code analysis, a graphical debugger, an integrated unit tester, integration with version control systems (VCSes), and supports web development with Django.

PyCharm is cross-platform, with Windows, macOS and Linux versions. The Community Edition is released under the Apache License, and there is also Professional Edition released under a proprietary license - this has extra features.

Features :

- Coding assistance and analysis, with code completion, syntax and error highlighting, linter integration, and quick fixes
- Project and code navigation: specialized project views, file structure views and quick jumping between files, classes, methods and usages
- Python refactoring: including rename, extract method, introduce variable, introduce constant, pull up, push down and others
- Support for web frameworks: Django, web2py and Flask
- Integrated Python debugger
- Integrated unit testing, with line-by-line code coverage
- Google App Engine Python development
- Version control integration: unified user interface for Mercurial, Git, Subversion, Perforce and CVS with changelists and merge

4.3 PACKAGES USED

4.3.1 OPENCV

OpenCV (Open Source Computer Vision) is a library of programming functions mainly aimed at real-time computer vision. The library is cross-platform and free for use under the open- source BSD license. OpenCV supports the deep learning frameworks TensorFlow, Torch/PyTorch and Caffe.

A Selection of Functionality:

- Image enhancement – Noise reduction, local contrast enhancement
    - Deconvolution – used to reduce focus blur or motion blur where the motion is known.
    - Unsharp masking – increases sharpness and local contrast
    - Histogram equalization – stretches contrast and somewhat corrects for over- or underexposure.
- Object classification and tracking – Track the paths that objects take in a scene. Differentiating between cars and trucks.
    - Foreground/background segmentation – identify objects moving in a scene.
    - Histogram backprojection – identify objects by their colour (even if they're not moving).
    - Camshift tracking – track objects by their colour.
- Face detection and recognition – Identify faces seen in images or video.
    - Detection: – Haar cascade – detect faces by identifying adjacent light and dark regions.
    - Recognition: – Eigenfaces classifier – for facial recognition

Advantages **:**

- First and foremost, OpenCV is available free of cost

- Since OpenCV library is written in C/C++ it is quite fast
- Low RAM usage (approx 60–70 mb)
- It is portable as OpenCV can run on any device that can run C

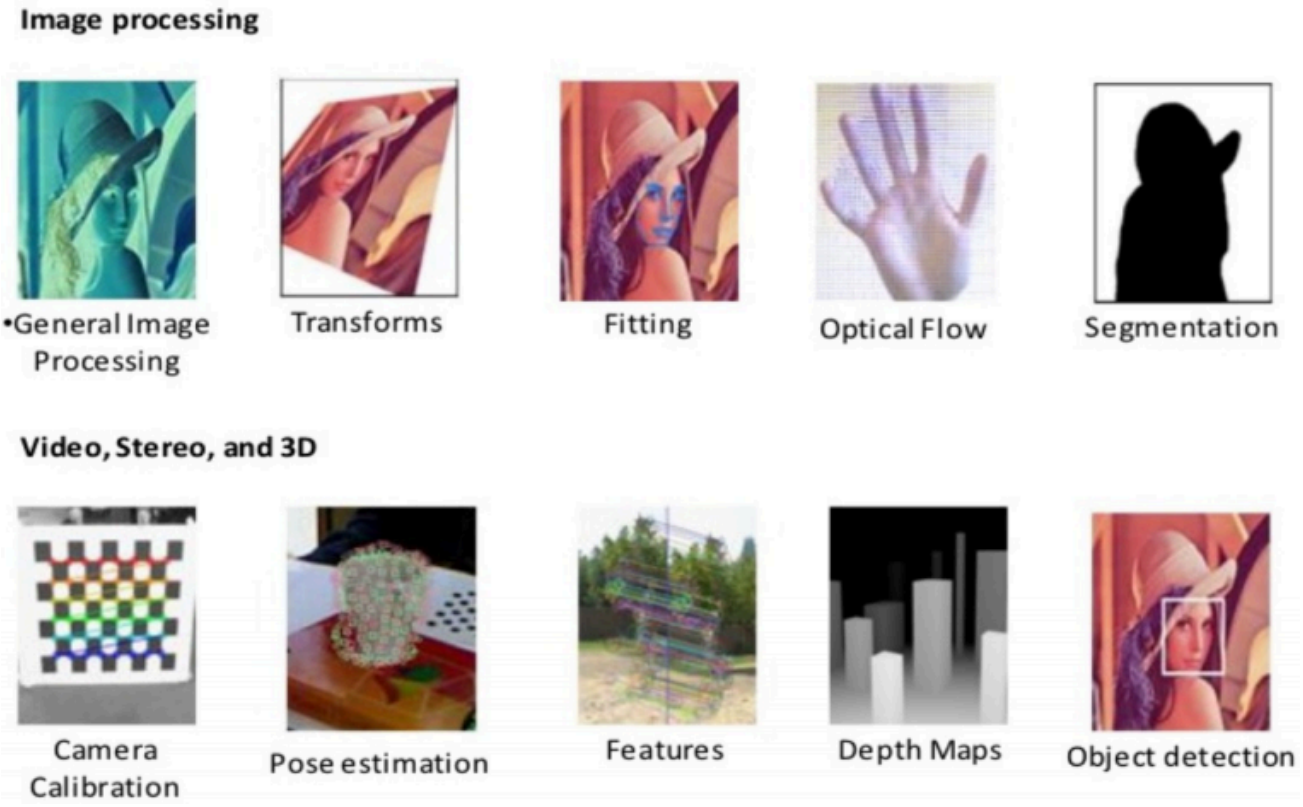


Fig3 : Package Capabilities

4.3.2 NUMPY

NumPy is a library for the Python programming language, adding support for large, multi dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.

The core functionality of NumPy is its "ndarray", for *n*-dimensional array, data structure. These arrays are strided views on memory. In contrast to Python's built-in list data structure (which, despite the name, is a dynamic array), these arrays are homogeneously typed: all elements of a single array must be of the same type.

Such arrays can also be views into memory buffers allocated by C/C++, Cython, and Fortran extensions to the CPython interpreter without the need to copy data around, giving a degree of compatibility with existing numerical libraries.

Advantages :

- NumPy is not just more efficient; it is also more convenient. You get a lot of vector and matrix operations for free, which sometimes allow one to avoid unnecessary work. And they are also efficiently implemented.
- NumPy array is faster and You get a lot built in with NumPy, FFTs, convolutions, fast searching, basic statistics, linear algebra, histograms, etc.

4.4 CONCEPTS USED

4.4.1 EDGE DETECTION

Edge detection includes a variety of mathematical methods that aim at identifying points in a digital image at which the image brightness changes sharply or, more formally, has discontinuities. The points at which image brightness changes sharply are typically organized into a set of curved line segments termed edge. Edge detection is a fundamental tool in image processing, machine vision and computer vision, particularly in the areas of feature detection and feature extraction.

The purpose of detecting sharp changes in image brightness is to capture important events and changes in properties of the world. It can be shown that under rather general assumptions for an image formation model, discontinuities in image brightness are likely to correspond to:

- discontinuities in depth
- discontinuities in surface orientation
- changes in material properties
- variations in scene illumination.

Canny Edge Detection

In this project we are using the Canny Edge Detection method**.** It is a multi-stage algorithm and we will go through each stages:

1. Noise Reduction
   Since edge detection is susceptible to noise in the image, the first step is to remove the noise in the image with a 5x5 Gaussian filter. We have already seen this in previous chapters.

2. Finding Intensity Gradient of the Image
   Smoothened image is then filtered with a Sobel kernel in both horizontal and vertical direction to get the first derivative in horizontal direction and vertical direction. Gradient direction is always perpendicular to edges. It is rounded to one of four angles representing vertical, horizontal and two diagonal directions.

3. Non-maximum Suppression
   After getting gradient magnitude and direction, a full scan of the image is done to remove any unwanted pixels which may not constitute the edge. For this, at every pixel, the pixel is checked if it is a local maximum in its neighborhood in the direction of gradient.

4.4.2 K-NEAREST NEIGHBORS ALGORITHM

In pattern recognition, the *k*-nearest neighbors algorithm (*k*-NN) is a non-parametric method used for classification and regression. In both cases, the input consists of the *k* closest training examples in the feature space. The output depends on whether *k*-NN is used for classification or regression:

- In k-NN classification, the output is a class membership. An object is classified by a majority vote of its neighbors, with the object being assigned to the class most common among its *k* nearest neighbors (*k* is a positive integer, typically small). If $k = 1$, then the object is simply assigned to the class of that single nearest neighbor.
- In k-NN regression, the output is the property value for the object. This value is the average of the values of its *k* nearest neighbors.
- KNN stores the entire training dataset which it uses as its representation.
- KNN does not learn any model.
- KNN makes predictions just-in-time by calculating the similarity between an input sample and each training instance.

k-NN is a type of instance-based learning, or lazy learning, where the function is only approximated locally and all computation is deferred until classification. The *k*-NN algorithm is among the simplest of all machine learning algorithms.

Both for classification and regression, a useful technique can be used to assign weight to the contributions of the neighbors, so that the nearer neighbors contribute more to the average than the more distant ones. For example, a common weighting scheme consists in giving each neighbor a weight of $1/d$, where d is the distance to the neighbor.

The neighbors are taken from a set of objects for which the class (for *k*-NN classification) or the object property value (for *k*-NN regression) is known. This can be thought of as the training set for the algorithm, though no explicit training step is required.

4.4.3 THRESHOLDING

- Thresholding converts to black and white image
- Colors are inverted, so (black,white) = (0,1)
- Some noise is present, but greatly reduced due to smoothing in previous step
- OpenCV: cv2.threshold()

4.4.4 MORPHOLOGICAL FILTERING

- Erosion: eat away at the boundaries of objects
- Removes white noise and small artifacts
- OpenCV: cv2.erode() with 4x4 kernel
- Dilation: increases thickness and white region opposite of erosion
- Useful in joining broken parts of an object
- OpenCV: cv2.dilate() with 2x2 kernel

4.4.5 OPENING VS CLOSING

- Erosion/Dilation = Opening

Eliminates noise outside objects

- Dilation/Erosion = Closing
  Eliminates noise within objects

4.4.6 CONTOUR/BOUNDING

- Box Contours = set of all outer contiguous points
- Approximate contours as a reduced polygon
- Calculate the bounding rectangle
- OpenCV: cv2.findContours(), cv2.approxPolyDP(), cv2.boundingRect()

4.4.7 KERNEL SMOOTHING

- Blurring helps eliminate noise after thresholding
- Local kernel defines averaging area
- Used 8x8 kernel for example images
- OpenCV: cv2.blur()

Chapter 5
TRAINING AND TESTING

TRAINING

Training is a very important process of working with a neural network. As seen from neural networks, there are two forms of training that can be employed with a neural network. They are namely:-

1. Un-Supervised Training 2. Supervised Training

Supervised training provides the neural network with training sets and the anticipated output. Unsupervised training supplies the neural network with training sets, but there is no anticipated output provided.

1. UNSUPERVISED TRAINING

Unsupervised training is a very common training technique for Kohonen neural networks. We will discuss how to construct a Kohonen neural network and the general process for training without supervision. 11 What is meant by training without supervision is that the neural network is provided with training sets, which are collections of defined input values. But the unsupervised neural network is not provided with anticipated outputs.

Unsupervised training is usually used in a classification neural network. A classification neural network takes input patterns, which are presented to the input neurons. These input patterns are then processed, and one single neuron on the output layer fires. This firing neuron can be thought of as the classification of which group the neural input pattern belonged to. Handwriting recognition is a good application of a classification neural network.

The input patterns presented to the Kohonen neural network are the dot image of the character that was hand written. We may then have 26 output neurons, which correspond to the 26 letters of the English alphabet. The Kohonen neural network should classify the input pattern into one of the 26 input patterns.

During the training process the Kohonen neural network in handwritten recognition is presented with 26 input patterns. The network is configured to also have 26 output patterns. As the Kohonen neural network is trained the weights should be adjusted so that the input patterns are classified into the 26 output neurons. This technique results in a relatively effective method for character recognition.

Another common application for unsupervised training is data mining. In this case you have a large amount of data, but you do not often know exactly what you are looking for. You want the neural network to classify this data into several groups. As the neural network trains the input patterns will fall into similar groups. This will allow you to see which input patterns were in common groups.

2. SUPERVISED TRAINING

The supervised training method is similar to the unsupervised training method in that training sets are provided. Just as with unsupervised training these training sets specify input signals to the neural network. The primary difference between supervised and unsupervised training is that in supervised training the expected outputs are provided. This allows the supervised training algorithm to adjust the weight matrix based on the difference between the anticipated output of the neural network, and the actual output. There are several popular training

algorithms that make use of supervised training. One of the most common is the back-propagation algorithm. It is also possible to use an algorithm such as simulated annealing or a genetic algorithm to implement supervised training.

TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of tests. Each test type addresses a specific testing requirement.

TYPES OF TESTS

Unit Testing
Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program input produces valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application. It is done after the completion of an individual unit before integration. This is a structural testing that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration Testing
Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfactory, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

System Testing
System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

Functional Testing
Functional tests provide a systematic demonstration that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.
Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures : interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identifying business process flows, data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

There are two basic approaches of functional testing:

1.Black box or functional testing. 2. White box testing or structural testing.

Black box testing

This method is used when knowledge of the specified function that a product has been designed to perform is known. The concept of black box is used to represent a system whose inside workings are not available for inspection. In a black box the test item is eaten as "Black", since its logic is unknown as to what goes in and what comes out, or the input and output. In black box testing, we try various inputs and examine the resulting outputs. The black box testing can also be used for scenarios based tests .In this test we verify whether it is taking valid input and producing resultant output to the user. It is imaginary box testing that hides internal workings. In our project valid input is image resultant output well structured image should be received.

White box testing

White box testing is concerned with testing implementation of the program. The intent of structural testing is not to exercise all the inputs or outputs but to exercise the different programming and data structures used in the program. Thus structure testing aims to achieve 59 test cases that will force the desired coverage of different structures.

RESULTS

UNIT TESTING

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Features to be tested

- Verify that the entries are of the correct format.
- No duplicate entries should be allowed.
- All links should take the user to the correct page.

INTEGRATION TESTING

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects. The task of the integration test is to check that components or software applications, ex. components in a software system or one step up software applications at the company level - interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

ACCEPTANCE TESTING

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

SUMMARY AND CONCLUSION

An Optical Character Recognising system has been developed which uses various knowledge sources to improve the performance. The composite characters are first segmented into its constituent symbol which helps in reducing the size of the set, in addition to being a natural way of dealing with different characters. The automated trainer makes two passes over the text image to learn the features of all the symbols of the text.

A character pair expert resolves confusion between two candidate characters. The composition processor puts the symbols back together to get the words which are then passed through the dictionary. The dictionary corrects only those characters which cause a mismatch and have been recognized with low confidence. The preliminary results on testing of the system shows performance of more than 93% on printed texts on individual fonts.

Further testing is currently underway for multi-font and hand printed texts. Most of the errors are due to inaccurate segmentation of symbols within a word. We are using only up to word level knowledge in our system. The domain knowledge and sentence level knowledge could be integrated to further enhance the performance in addition to making it more robust.

The method utilizes an initial stage in which successive columns (vertical strips) of the scanned array are ORed in groups of one pitch width to yield a coarse line pattern (CLP) that crudely shows the distribution of white and black along the line.

The CLP is analyzed to estimate baseline and line skew parameters by transforming the CLP by different trial line skews within a specified range. For every transformed CLP (XCLP), the number of black elements in each row is counted and the row-to-row change in this count is also calculated. .The XCLP giving the maximum negative change (decrease) is assumed to have zero skew. The skew corrected row that gives the maximum gradient serves as the estimated baseline. Successive pattern fields of the scanned array having unit pitch width are superposed (after skew correction) and summed.

The resulting sum matrix tends to be sparse in the inter-character area. Thus, the column having minimum sum is recorded as an "average", or coarse, X-direction segmentation position. references.

Each character pattern is examined individually, with the known baseline (corrected for skew) and average segmentation column as references. A number of neighboring columns (3 columns, for example) to the left and right of the average segmentation columns are included in the view that is analyzed for full segmentation by conventional algorithm.

Various algorithms and techniques for pre-processing and character recognition from a image and implemented most optimal ones amongst them, thus resulting in more speed and accuracy. This makes the project dynamic and is feasible for any kind of organization.

REFERENCES

1. https://en.wikipedia.org/wiki/Optical_character_recognition
2. http://www.cvisiontech.com/library/ocr/ocr-pdf/ocr-software-handwriting-recognition.html
3. https://www.learnpick.in/prime/documents/ppts/details/1246/optical-character-recognition
4. http://www.assistivetechnology.vcu.edu/wpcontent/uploads/sites/1864/2013/09/pxc3882784.pdf
5. https://www.codeproject.com/Articles/15304/Unicode-Optical-Character-Recognition
6. https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_canny/py_canny.html
7. https://www.pyimagesearch.com/2017/07/10/using-tesseract-ocr-python/
8. https://www.ocr.org.uk/qualifications/by-subject/computing/computing-resources/databases/
9. http://www.legalscans.com/ocr.html
10. https://searchcontentmanagement.techtarget.com/definition/OCR-optical-character-recognition

APPENDIX A

SCREENSHOTS

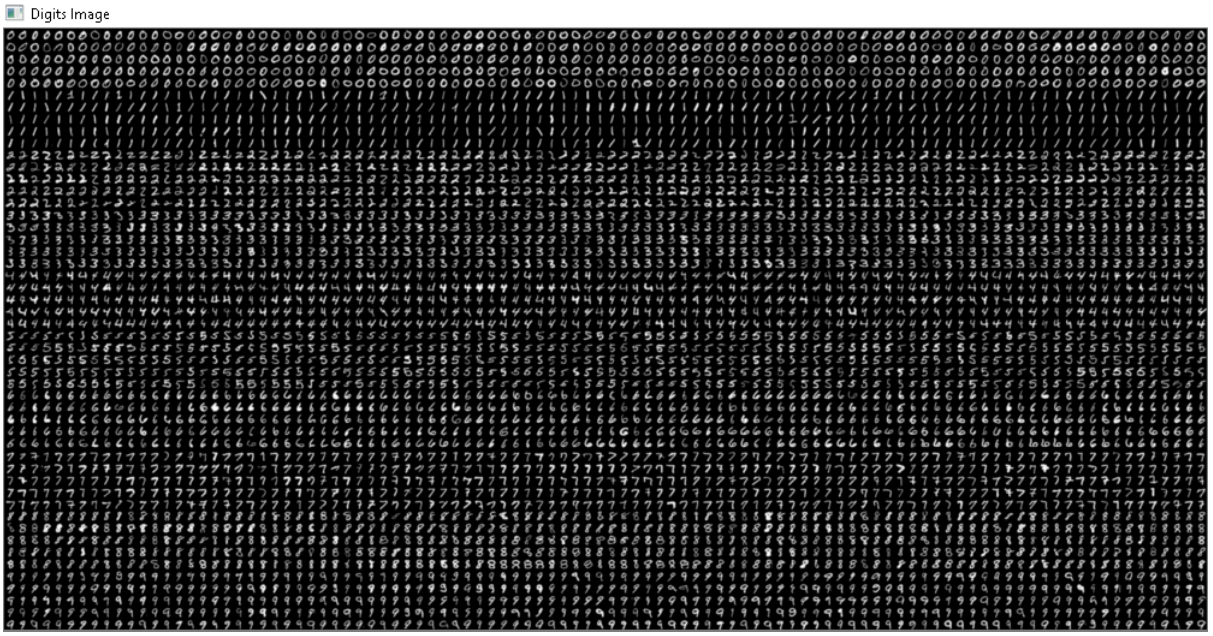


Fig4 : Characters Database

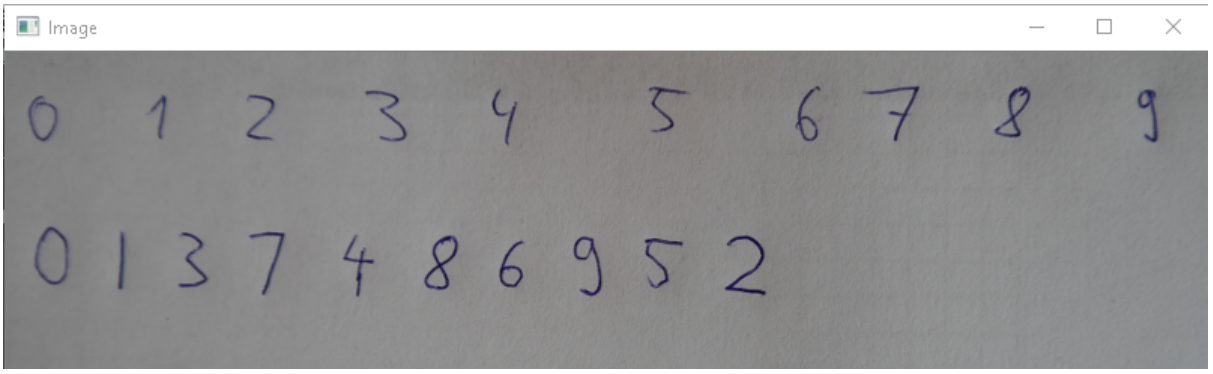


Fig5 : Handwritten Image of Digits

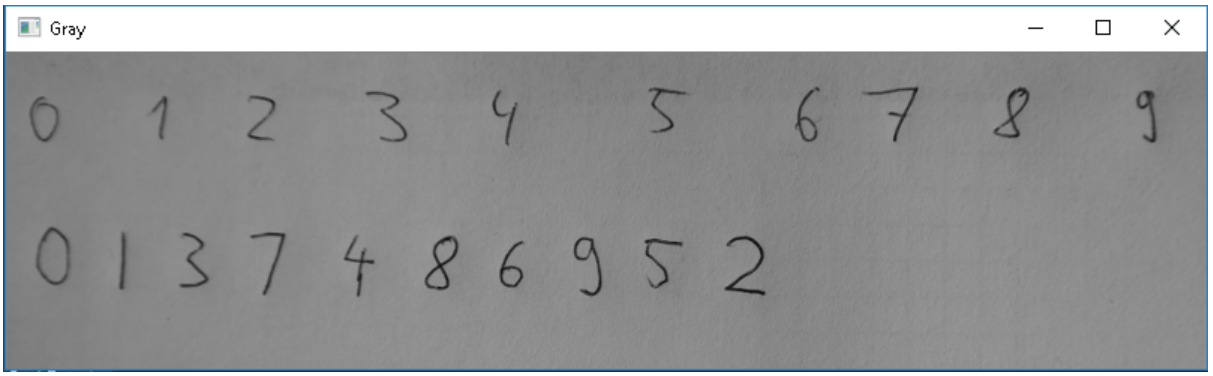


Fig6 : Conversion into GrayScale Image

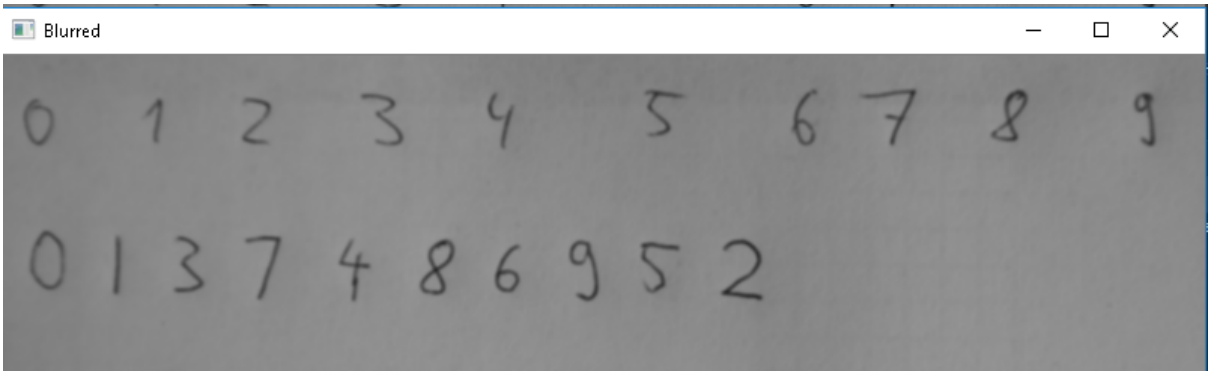


Fig7 : Conversion into Blurred Image

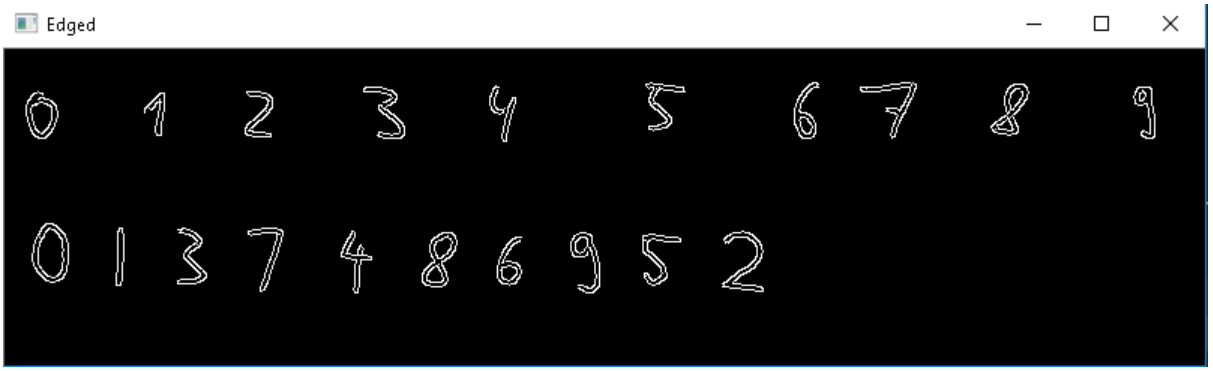


Fig8 : Edge Detection of the Image

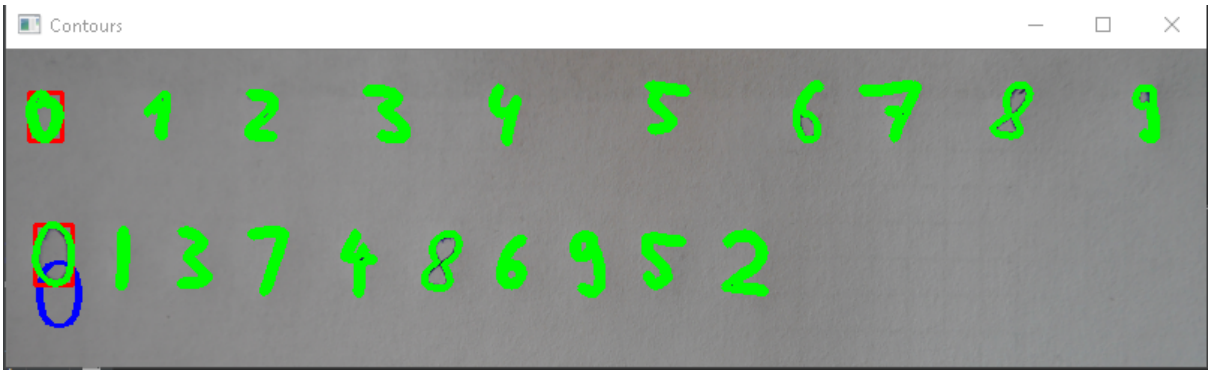


Fig9 : Contour/Bounding

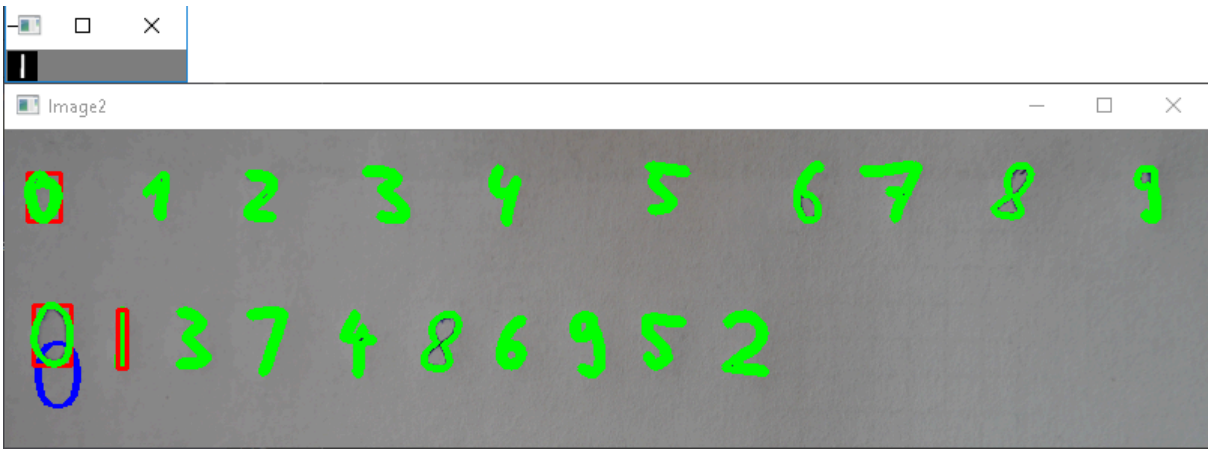


Fig10 : Character Detection and Display

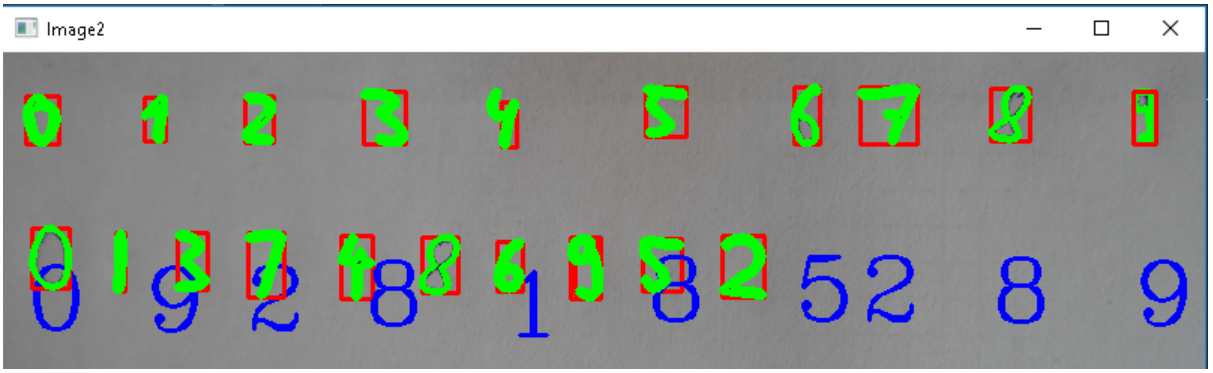


Fig11 : Final Output

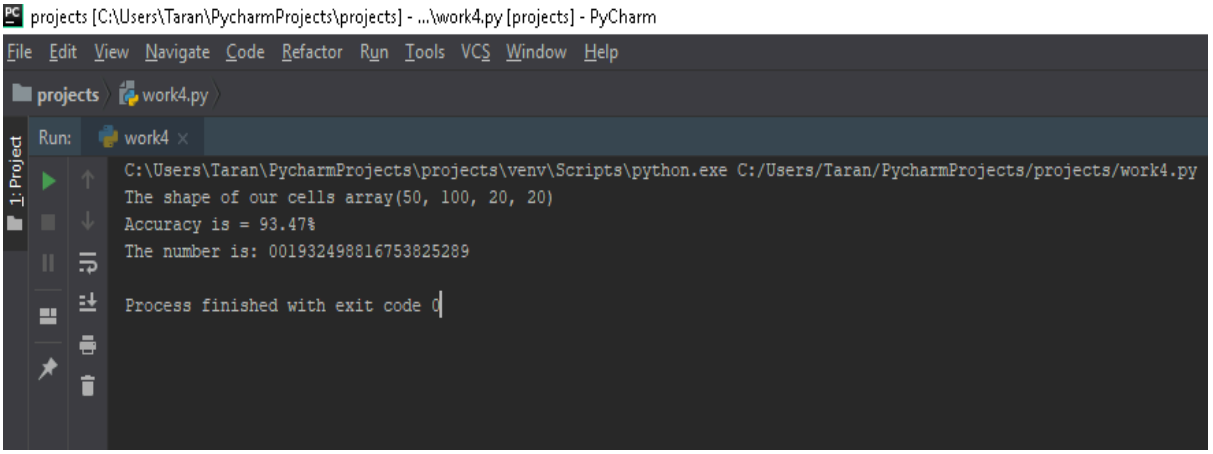


Fig12 : Complete Detection of Image Text