

A
Project Report
On

ABNORMAL ACTIVITY DETECTION USING CONVOLUTIONAL NEURAL NETWORKS

Submitted for partial fulfillment of requirements for the award of degree of
**BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE AND
ENGINEERING BY**

Mr. YASWANT MEKA 16K81A0560

UNDER THE SUPERVISION OF

Dr. T. Poongothai

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



St.Martin's Engineering College
(Affiliated to JNTU,Hyderabad)

Dhullapally(V),Qutubullapur(M),Secunderabad

2019-2020



ST.MARTIN'S ENGINEERING COLLEGE

An Autonomous Institute

Dhulapally, Secunderabad -500100



NAAC A+ Accredited

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the work embodies in this dissertation entitled “**Abnormal Activity Detection Using Convolutional Neural Network**” is being submitted by
Mr. YASWANT MEKA(16K81A0560) for partial fulfillment of the requirement for the award of degree of **Bachelor Of Technology in Computer Science And Engineering** discipline to **ST.MARTIN'S ENGINEERING COLLEGE DHULAPALLY, KOMPALLY, SECUNDERABAD (T.S)** during the academic year 2019-2020 is record of bonafide piece of work, undertaken by him/her the supervision of undersigned.

Internal Guide

Dr. T. Poongothai

Head of the Department

Dr. P.Udayakumar

Internal Examiner

External Examiner

ST.MARTIN'S ENGINEERING COLLEGE

An Autonomous Institute



Dhulapally, Secunderabad-500100



NAAC A+ Accredited

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING

DECLARATION

Myself 'YASWANT' as a student of

'Bachelor of technology in Computer Science and Engineering', session: 2019/2020,

St.martins engineering college, Dhullapally, Kompally, Secunderabad,

Telangana State, hereby declare that the work presented in this project work entitled

'Abnormal Activity Detection Using Convolutional Neural Network ' is the

outcome of our own bonafide work and is correct to the best of our knowledge and

this work has been undertaken taking care of Engineering Ethics. It contains no

material previously published or written by another person nor material which has

been accepted for the award of any other degree or diploma of university or other

institute of higher learning, except where due acknowledgement has been made in

the text.

Date:

Mr. M. Yaswant (16K81A0560)

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompanies the successful completion of any task would be incomplete without the mention of the people who made it possible and whose encouragement and guidance have crowded our efforts with success.

We extend our deep sense of gratitude to Principal, **Dr. P. SANTOSH KUMAR PATRA**, St. Martin's Engineering College, Dhulapally, for permitting us to undertake this project.

We are also thankful to **Dr.P.UDAYAKUMAR**, Head of the Department, Computer Science and Engineering, St. Martin's Engineering College, Dhulapally, for his support and guidance throughout our project, and as well as our Project Coordinator **Dr.P.K.A.Chitra**, Associate Professor, in Computer Science and Engineering department for his valuable support.

We would like to express our sincere gratitude and indebtedness to our project supervisor **Dr.Poongothai**, Professor, Computer Science and Engineering, St. Martin's Engineering College, Dhulapally, for his support and guidance throughout our project.

Finally, we express thanks to all those who have helped us successfully to completing this project. Furthermore, we would like to thank our family and friends for their moral support and encouragement. We express thanks to all those who have helped us in successfully completing the project.

Mr. M. Yaswant

16K81A0560

ABSTRACT

Abnormal Activities Detection (AAD) is a widely studied computer vision problem. Applications of AAD include images like health care and human-computer interaction. As the imaging technique advances upgrades, novel approaches for AAD constantly emerge. Abnormal Activities Detection is a significant component of many innovative and human-behavior based systems. The ability to recognize various human activities enables the developing of intelligent control system. Usually the task of Abnormal Activities Detection is mapped to the classification task of images representing person's actions. This Project addresses the problem of human activities' classification using deep learning methods such as Convolutional Neural Networks. Deep learning architectures such as deep neural networks, deep belief networks, recurrent neural networks and convolutional neural networks have been applied to fields including computer vision, speech recognition, natural language processing, audio recognition, social network filtering, machine translation, bioinformatics, drug design, medical image analysis, material inspection and board game programs, where they have produced results comparable to and in some cases surpassing human expert performance. A novel pooling mechanism called rank-based pooling is proposed. We consider rank-based pooling as an instance of weighted pooling where a weighted sum of activations is used to generate the pooling output. Image or Object Detection is a computer technology that processes the image and detects objects in it. People often confuse Image Detection with Image Classification. Although the difference is rather clear. If you need to classify image items, you use Classification. But if you just need to locate them, for example, find out the number of objects in the picture, you should use Image Detection. Fire-Detection-Image-Dataset is one of the dataset contains normal images and images with fire. It is highly unbalanced to reciprocate real world situations. It consists of a variety of scenarios and different fire situations (intensity, luminosity, size, environment etc).

This Project provides the comparison study on these methods applied for Abnormal Activities Detection task using the set of images representing five different categories of daily life activities like Handguns, Knives, Running.

LIST OF CONTENTS

1. INTRODUCTION	9
1.1 INTRODUCTION	9
1.2 PURPOSE OF THE PROJECT	9
1.3 PROBLEMS IN EXISTING SYSTEM	10
2. LITERATURE REVIEW	11
2.1 LITERATURE SURVEY	11
2.2 EXISTING SYSTEM	12
2.3 PROPOSED SYSTEM	13
3. METHODOLOGY	15
3.1 SYSTEM ANALYSIS	15
3.2 FUNCTIONAL REQUIREMENTS	16
3.3 NON-FUNCTIONAL REQUIREMENTS	17
3.4 HARDWARE AND SOFTWARE REQUIREMENTS	18
3.5 SYSTEM ARCHITECTURE	18
3.6 UML DIAGRAMS	19
4. SYSTEM IMPLEMENTATION	31
4.1 PYTHON	31
4.2 SAMPLE CODE	32
4.3 DATASET	40
4.4 TESTING	42
5.RESULTS AND CONCLUSION	49
REFERENCES	53

LIST OF FIGURES

Figure number	Fig name	Page no
Fig-2.1	model of convolutional neural network	13
Fig-3.1	SDLC Framework	15
Fig-3.2	System Architecture	19
Fig-3.3	Use case diagram	27
Fig-3.4	Sequence diagram	28
Fig-3.5	Activity diagram	29
Fig-3.6	component diagram	30
Fig-3.7	Python deployment	30

LIST OF TABLES

Table number	Table name	Page no
Tab.no-4.1	Testcases	48

LIST OF ABBREVIATIONS

1. CNN- Convolutional Neural Network
2. AAD- Abnormal Activity Detection
3. SDLC- Software Development Life Cycle
4. ReLU- Rectified Linear Unit
5. CSV- Comma Seperated Values
6. IDE- Integrated Development Environment

CHAPTER 1-INTRODUCTION

1.1 INTRODUCTION

The idea of activity recognition in pervasive computing is of utmost importance because of its varied applications in real-life, particularly to handle human-centric problems such as security. The purpose of an activity recognition system is to recognize the basic daily life activities of human beings. Due to the diversity and complexity in human activities, the accuracy of human action recognition becomes challenging. Construction of activity models that identify and classify various human activities follow several approaches. This research discipline attracts video processing and machine learning communities as it finds applications in several fields of studies like medicine and healthcare, human-computer interaction, crime investigation and security systems. Applications of human activity recognition are not limited to health care and security. Human activity recognition system uses sensors for interpretation of gestures or motion of the human being, thus identifying the action that the human body makes. Understanding human activities hold within itself a recognition of the activity and pattern discovery of that activity. The initial part makes use of a predefined activity model to correctly detect human activity. Thus, there is a need to construct a high-level conceptual model for implementing the pervasive system for human activity recognition. Moreover, activity pattern discovery does not require predefined models as it uses only a few low-level sensor data that is captured in order to find the unknown patterns. Though there is a major difference in the two techniques they have a common target of improving the performance of human activity recognition systems. These techniques are also supportive of each other. They combine to enhance the performance by using the discovery of the activity pattern to define recognized activity. Abnormal activity recognition is considered as the most challenging task from images. Due to the

traditional method depend on the computation of artificial features, and noise data has some influence on the extracted features.

1.2 PURPOSE OF THE PROJECT

In this paper, a new hybrid deep learning structure was proposed to fuse the extracted features, which integrates convolutional neural network (CNN) and long short-term memory network (LSTM). Firstly, the video was preprocessed and extracted visual features by CNN. Next, LSTM was used to learn the temporal features of visual features and added attention mechanism to select important features. Finally, the video feature vector obtained layer by layer to judge abnormal activity.

An experiment is used to test the ability of the model on the standard dataset to recognize abnormal activity, the result shows that our experimental demonstrate high performance of recognition and outperform the state-of-art algorithms.

1.3 PROBLEMS IN EXISTING SYSTEM

In computer vision one of the major research areas is on surveillance images. In surveillance Image, real time detection of abnormal events at the time of happening is one of the major challenging task. Abnormal means deviating from what is normal or uncommon behavior.

Consider any public places like Airports, Bus stands, Railway stations, Temples , market, plaza and traffic place where unexpected events or abnormal events happens like fires, explosions, accidents, transportation disasters etc. At that time people scared and would trying to escape from those places. Feature extraction is the process of selection of the variables or attributes which has some co relation with the class of that instance. Features are classified into two categories. First one is local features, it can also be referred as structural feature. These features contains structural elements like loop, branches, crossing point, joints, points, curve etc. Second one is global features, they can be computed on image pattern and they includes the features like histogram, projection, distance, etc.

So that by taking crowds behavior before and after the event happened, we can detect abnormal events. For controlling the huge damage to the society and public safety, detection of abnormal events at the time of happening is one of the useful technologies. Hence semi/fully automated intelligent systems are required to intimate the man kind at the time of happening will lead to controlling the huge damage to the society. It can be used in finding the terror attack, Bomb placement, religion disputes, attacks,

accidents, violation of traffic rules etc. Here our objective is to develop more robust system to handle various anomaly and help the surveillance system.

CHAPTER 2-LITERATURE REVIEW AND PROBLEM IDENTIFICATION

2.1 LITERATURE SURVEY

- Extensive literature has been produced about sensor-based activity recognition. Bulling et al. [18] give a broad introduction to the problem, highlighting the capabilities and limitations of the classification models based on static and shallow features. Alsheikh et al. [7] introduce a first approach to HAR based on deep learning models. They generate a spectrogram image from an inertial signal, in order to feed real images to a convolutional neural network. This approach overcomes the need for reshaping the signals in a suitable format for a CNN, however, the spectrogram generation step simply replaces the process of feature extraction, adding initial overhead to the network training. [8]
- Zeng et al. use raw acceleration signals as input for a convolutional network [16], applying 1-D convolution to each signal component. This approach may result in loss of spatial dependencies among different components of the same sensor. They focus on public datasets, obtained mainly from embedded sensors (like smartphones), or worn sensors placed on the arm. A similar technique is suggested by
- Yang et al. In their work, they use the same public datasets, however, they apply 2-D convolution over a single-channel representation of the kinetic signals. This particular application of CNNs for the activity recognition problem is further elaborated by
- Ha et al., with a multi-channel convolutional network that leverages both acceleration and angular velocity signals to classify daily activities from a public dataset of upper-limb movements. The classification task they perform is personalized, so the signals gathered from each participant are

used to train individual learning models. One of the missing elements in all the previously described contributions about deep learning models is a comparison of the classification performance of individual sensors or group of sensors.

- Fire and smoke detection with Keras and Deep Learning by Adrian Rosebrock on November 18, 2019 [2] stated that the dataset they used for fire and smoke examples was curated by PyImageSearch reader, Gautam Kumar. Guatam gathered a total of 1,315 images by searching Google Images for queries related to the term “fire”, “smoke”, etc. However, the original dataset has not been cleansed of extraneous, irrelevant images that are not related to fire and smoke (i.e., examples of famous buildings before a fire occurred).
- Rank-based pooling for deep convolutional neural networks by ZenglinShi , YangdongYe, YunpengWu [20] approach shows that pooling is a key mechanism in deep convolutional neural networks (CNNs) which helps to achieve translation invariance. Numerous studies, both empirically and theoretically, show that pooling consistently boosts the performance of the CNNs. The conventional pooling methods are operated on activation values. In this work, we alternatively propose rank-based pooling. It is derived from the observations that ranking list is invariant under changes of activation values in a pooling region, and thus rank-based pooling operation may achieve more robust performance.
- Our aim in this paper is to implement a deep CNN that can properly address the task of activity recognition, and then compare the results obtained with the adoption of different sensor combinations. We also focus on a set of exercise activities that are part of the Otago exercise program. To the best of our knowledge, this group of activities has never been explored before in the context of activity recognition.

2.2 EXISTING SYSTEM

The System of existing literature focuses on finding patterns in passenger activity records.

Such knowledge can be useful in a variety of applications, and plays a vital role in effectively finding and satisfying passenger needs.

Examples include assessing the performance of the transit network, identifying and optimizing problematic or flawed bus routes, improving the accuracy of passenger flow forecasted between two regions and making service adjustments that accommodate variations in ridership on different days. In

particular, estimated the crowdedness of various stations in the transportation network using AFC data measured the variability of transit behaviors on different days of the week.

Existing studies that detect anomalies in urban sensing data can be divided into two categories: those based on locations, and those on trajectories.

Along the line of location-based anomaly detection, presented a framework that learned the context of different functional regions in a city, which provided the basis of our feature extraction approach.

2.3 PROPOSED SYSTEM

Convolution Neural Networks or convNets are neural networks that share their parameters.[5] Types of layers:

- Input Layer
- Convolution Layer
- Activation Function Layer
- Pool Layer
- Fully-Connected Layer

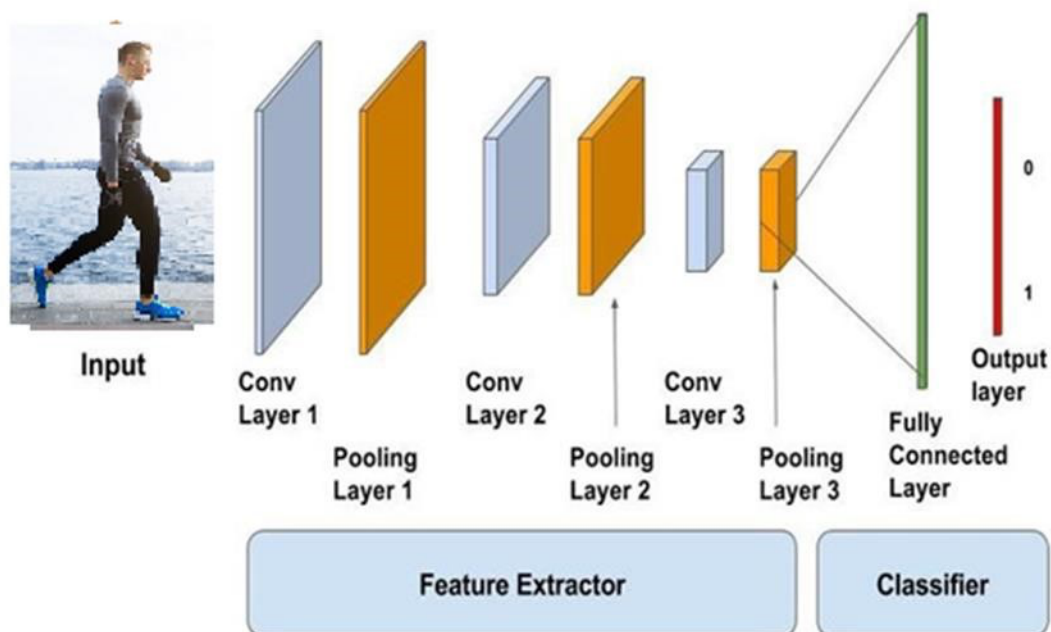


Fig 2.1 model of convolutional neural network

The above figure describes the model of convolutional neural network process.

Input Layer: This layer holds the raw input of image with width 32, height 32 and depth 3.

Convolution Layer: This layer computes the output volume by computing dot product between all filters and image patch. Suppose we use total 12 filters for this layer we'll get output volume of dimension $32 \times 32 \times 12$.

Activation Function Layer: This layer will apply element wise activation function to the output of convolution layer. Some common activation functions are RELU: $\max(0, x)$, Sigmoid: $1/(1+e^{-x})$, Tanh, Leaky RELU, etc.[14]

Convolutional Layer: Creates a feature map to predict the class probabilities for each feature by applying a filter that scans the whole image, few pixels at a time.

Pooling Layer (down sampling): scales down the amount of information the convolutional layer generated for each feature and maintains the most essential information (the process of the convolutional and pooling layers usually repeats several times).[1]

Fully Connected Input Layer: “flattens” the outputs generated by previous layers to turn them into a single vector that can be used as an input for the next layer.

Fully Connected Layer: applies weights over the input generated by the feature analysis to predict an accurate label.

Fully Connected Output Layer: generates the final probabilities to determine a class for the image.

CHAPTER 3- METHODOLOGY

3.1 SYSTEM ANALYSIS

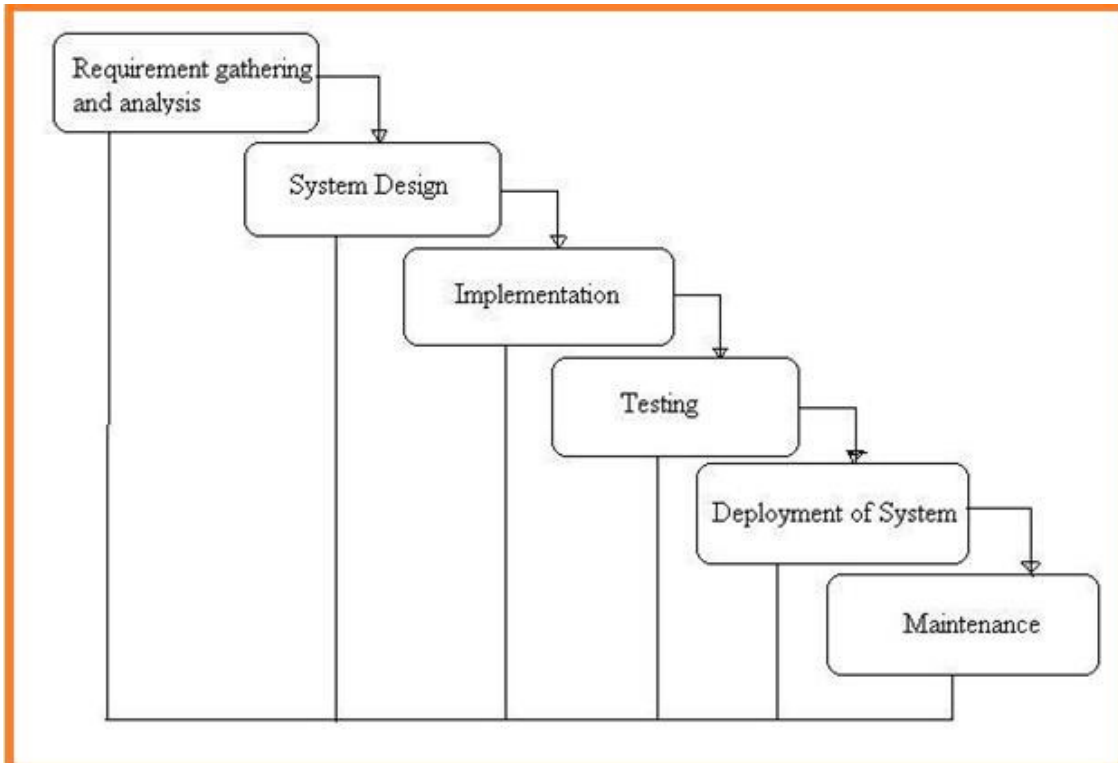


Fig 3.1 Software Development Life Cycle Framework

Waterfall approach was first SDLC Model to be used widely in Software Engineering to ensure success of the project. In "The Waterfall" approach, the whole process of software development is divided into separate phases. In this Waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially.

The following illustration is a representation of the different phases of the Waterfall Model

- **Requirement Gathering and analysis** – All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.

- **System Design** – The requirement specifications from first phase are studied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.
- **Implementation** – With inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.
- **Integration and Testing** – All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.
- **Deployment of system** – Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.
- **Maintenance** – There are some issues which come up in the client environment. To fix those issues, patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

3.2 FUNCTIONAL REQUIREMENTS

- Admin Login
- Prediction
- Training
- Testing Image
- Result message

Admin Login

Authentication and Authorization Authentication means confirming your own identity, whereas authorization means being allowed access to the system. In even more simpler terms authentication is the process of verifying oneself, while authorization is the process of verifying what you have access to any application

Training

Training a simple Convolutional Neural Network (CNN) to classify Activity images. Because this tutorial uses the Keras Sequential API [17], creating and training our model will take just a few lines of code. The image dataset contains 1000 color images in 2 classes, with 1000 images in

each class. The dataset is divided into 1000 training images and 1000 testing images. The classes are mutually exclusive and there is no overlap between them.

Prediction

The CNN model will learn a function that maps a sequence of past observations as input to an output observation. We can divide the sequence into multiple input/output patterns called samples, where three time steps are used as input and one time step is used as output for the onestep prediction that is being learned.[15]

Testing Image

In this module user will give different activity images as image upload from this it will process from our “weights.h5” and created classifier we will get the output of user activity normal activity & abnormal activity

Result message

In this module it will show the results from the test images output of user activity normal activity & abnormal activity.

3.3 NON-FUNCTIONAL REQUIREMENTS

Expanded System admin security: overseer to eschew the abuse of the application by PC ought to be exceptionally secured and available.

Compactness: The Presentation of this application is facile to utilize so it is looks simple for the using client to comprehend and react to identically tantamount.

Unwavering quality: and the functionalities accessible in the application thissubstructure has high probability to convey us the required inquiries.

Time take for Reaction: The time taken by the application to culminate an undertaking given by the client is very fast.

Multifariousness: Our application can be stretched out to incorporate the vicissitudes done by applications present now to enhance the performance of the item. This is implicatively insinuated for the future works that will be done on the application.

Vigor: The project is blame tolerant concerning illicit client/beneficiary sources of info. Blunder checking has been worked in the platforms to avert platforms disappointment.

3.4 SYSTEM CONFIGURATION:

Hardware requirements:

Processor	:	Any Update Processor
Ram	:	Min 1 GB Hard
Disk	:	Min 100 GB

Software requirements:

Operating System	:	Windows family
Technology	:	Python 3.6
IDE	:	PyCharm

3.5 SYSTEM ARCHITECTURE

A convolutional layer within a neural network should have the following attributes:

- Convolutional kernels defined by a width and height (hyper-parameters).
- The number of input channels and output channels (hyper-parameter).
- The depth of the Convolution filter (the input channels) must be equal to the number channels (depth) of the input feature map convolution.

There are two main parts to a CNN:

A convolution tool that splits the various features of the image for analysis.

A fully connected layer that uses the output of the convolution layer to predict the best description for the image.

The neurons within a CNN are split into a three-dimensional structure, with each set of neurons analyzing a small region or feature of the image. In other words, each group of neurons specializes in identifying one part of the image. CNNs use the predictions from the layers to produce a final output that presents a vector of probability scores to represent the likelihood that a specific feature belongs to a certain class.[3]

When programming a CNN, the input is a tensor with shape :

(number of images) x (image width) x (image height) x (image depth).

Then after passing through a convolutional layer, the image becomes abstracted to a feature map, with shape :

(number of images) x (feature map width) x (feature map height) x (feature map channels) ReLU:
[13]

Rectified linear unit is an activation function defined as the positive part of its argument: $f(x)=\max(0,x)$ where x is the input to a neuron.

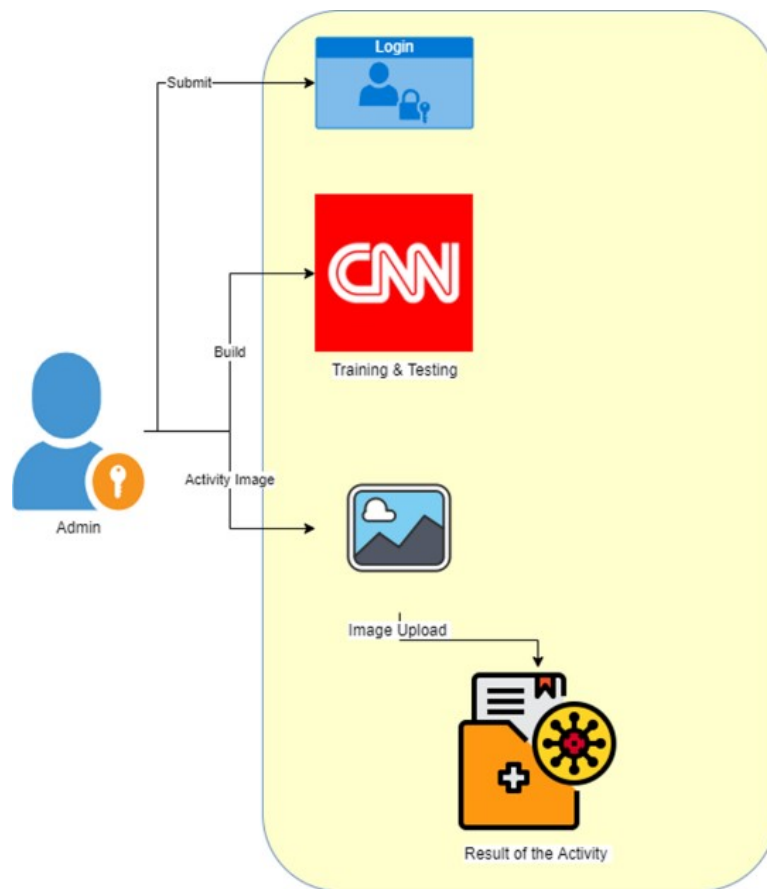


Fig 3.2 system architecture

3.6 UML DIAGRAMS

The System Design Document describes the system requirements, operating environment, system and subsystem architecture, files and database design, input formats, output layouts, human-machine interfaces, detailed design, processing logic, and external interfaces.

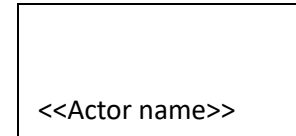
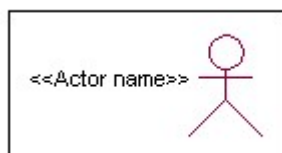
Global Use Case Diagrams:

Identification of actors:

Actor: Actor represents the role a user plays with respect to the system. An actor interacts with, but has no control over the use cases.



Actor



Graphical representation:

An actor is someone or something that:

Interacts with or uses the system.

- Provides input to and receives information from the system.
- Is external to the system and has no control over the use cases.

Actors are discovered by examining:

- Who directly uses the system?
- Who is responsible for maintaining the system?
- External hardware used by the system.
- Other systems that need to interact with the system.

Questions to identify actors:

- Who is using the system? Or, who is affected by the system? Or, which groups need help from the system to perform a task?
- Who affects the system? Or, which user groups are needed by the system to perform its functions? These functions can be both main functions and secondary functions such as administration.
- Which external hardware or systems (if any) use the system to perform tasks?
- What problems does this application solve (that is, for whom)?
- And, finally, how do users use the system (use case)? What are they doing with the system?

The actors identified in this system are:

- a. System Administrator**
- b. Customer**
- c. Customer Care**

Identification of usecases:

Usecase: A use case can be described as a specific way of using the system from a user's (actor's) perspective.

Graphical representation:



A more detailed description might characterize a use case as:

- Pattern of behavior the system exhibits
- A sequence of related transactions performed by an actor and the system
- Delivering something of value to the actor Use cases provide a means to:
- capture system requirements

- communicate with the end users and domain experts
- test the system

Use cases are best discovered by examining the actors and defining what the actor will be able to do with the system.

Guide lines for identifying use cases:

- For each actor, find the tasks and functions that the actor should be able to perform or that the system needs the actor to perform. The use case should represent a course of events that leads to clear goal
- Name the use cases.
- Describe the use cases briefly by applying terms with which the user is familiar.

This makes the description less ambiguous Questions

to identify use cases:

- What are the tasks of each actor?
- Will any actor create, store, change, remove or read information in the system?
- What use case will store, change, remove or read this information?
- Will any actor need to inform the system about sudden external changes?
- Does any actor need to inform about certain occurrences in the system?
- What usecases will support and maintains the system?

Flow of Events

A flow of events is a sequence of transactions (or events) performed by the system. They typically contain very detailed information, written in terms of what the system should do, not how the system accomplishes the task. Flow of events are created as separate files or documents in your favorite text editor and then attached or linked to a use case using the Files tab of a model element.

A flow of events should include:

- When and how the use case starts and ends
- Use case/actor interactions
- Data needed by the use case
- Normal sequence of events for the use case
- Alternate or exceptional flows

Construction of Usecase diagrams:
Use-case diagrams graphically depict system behavior (use cases). These diagrams present a high level view of how the system is used as viewed from an outsider's (actor's) perspective. A use-case diagram may depict all or some of the use cases of a system.

A use-case diagram can contain:

- actors ("things" outside the system)
- use cases (system boundaries identifying what the system should do)
- Interactions or relationships between actors and use cases in the system including the associations, dependencies, and generalizations.

Relationships in use cases:

1. Communication:

The communication relationship of an actor in a usecase is shown by connecting the actor symbol to the usecase symbol with a solid path. The actor is said to communicate with the usecase.

2. Uses:

A Uses relationship between the usecases is shown by generalization arrow from the usecase.

3. Extends:

The extend relationship is used when we have one usecase that is similar to another usecase but does a bit more. In essence it is like subclass.

SEQUENCE DIAGRAMS

A sequence diagram is a graphical view of a scenario that shows object interaction in a time-based sequence what happens first, what happens next. Sequence diagrams establish the roles of objects and help provide essential information to determine class responsibilities and interfaces.

There are two main differences between sequence and collaboration diagrams: sequence diagrams show time-based object interaction while collaboration diagrams show how objects associate with each other. A sequence diagram has two dimensions: typically, vertical placement represents time and horizontal placement represents different objects.

Object:

An object has state, behavior, and identity. The structure and behavior of similar objects are defined in their common class. Each object in a diagram indicates some instance of a class. An object that is not named is referred to as a class instance.

The object icon is similar to a class icon except that the name is underlined:

An object's concurrency is defined by the concurrency of its class.

Message:

A message is the communication carried between two objects that trigger an event. A message carries information from the source focus of control to the destination focus of control. The synchronization of

a message can be modified through the message specification. Synchronization means a message where the sending object pauses to wait for results.

Link:

A link should exist between two objects, including class utilities, only if there is a relationship between their corresponding classes. The existence of a relationship between two classes symbolizes a path of communication between instances of the classes: one object may send messages to another. The link is depicted as a straight line between objects or objects and class instances in a collaboration diagram. If an object links to itself, use the loop version of the icon.

CLASS DIAGRAM:

Identification of analysis classes:

A class is a set of objects that share a common structure and common behavior (the same attributes, operations, relationships and semantics). A class is an abstraction of real-world items.

There are 4 approaches for identifying classes:

- a. Noun phrase approach:
- b. Common class pattern approach.
- c. Use case Driven Sequence or Collaboration approach.
- d. Classes , Responsibilities and collaborators Approach

1. Noun Phrase Approach:

The guidelines for identifying the classes:

- Look for nouns and noun phrases in the usecases.
- Some classes are implicit or taken from general knowledge.
- All classes must make sense in the application domain; Avoid computer implementation classes – defer them to the design stage.
- Carefully choose and define the class names After identifying the classes we have to eliminate the following types of classes:
- Adjective classes.

2. Common class pattern approach:

The following are the patterns for finding the candidate classes:

- Concept class.
- Events class.
- Organization class
- Peoples class

- Places class
- Tangible things and devices class.

3. Use case driven approach:

We have to draw the sequence diagram or collaboration diagram. If there is need for some classes to represent some functionality then add new classes which perform those functionalities.

4. CRC approach:

The process consists of the following steps:

- Identify classes' responsibilities (and identify the classes)
- Assign the responsibilities
- Identify the collaborators.

Identification of responsibilities of each class:

The questions that should be answered to identify the attributes and methods of a class respectively are:

- a. What information about an object should we keep track of?
- b. What services must a class provide?

Identification of relationships among the classes:

Three types of relationships among the objects are:

Association: How objects are associated?

Super-sub structure: How are objects organized into super classes and sub classes?

Aggregation: What is the composition of the complex classes?

Association:

The questions that will help us to identify the associations are:

- a. Is the class capable of fulfilling the required task by itself?
- b. If not, what does it need?
- c. From what other classes can it acquire what it needs?

Guidelines for identifying the tentative associations:

- A dependency between two or more classes may be an association. Association often corresponds to a verb or prepositional phrase.
- A reference from one class to another is an association. Some associations are implicit or taken from general knowledge.

Some common association patterns are:

Location association like part of, next to, contained in.....

Communication association like talk to, order to

We have to eliminate the unnecessary association like implementation associations, ternary or n-ary associations and derived associations.

Super-sub class relationships:

Super-sub class hierarchy is a relationship between classes where one class is the parent class of another class (derived class). This is based on inheritance.

Guidelines for identifying the super-sub relationship, a generalization are

1. Top-down:

Look for noun phrases composed of various adjectives in a class name. Avoid excessive refinement. Specialize only when the sub classes have significant behavior.

2. Bottom-up:

Look for classes with similar attributes or methods. Group them by moving the common attributes and methods to an abstract class. You may have to alter the definitions a bit.

3. Reusability:

Move the attributes and methods as high as possible in the hierarchy.

4. Multiple inheritances:

Avoid excessive use of multiple inheritances. One way of getting benefits of multiple inheritances is to inherit from the most appropriate class and add an object of another class as an attribute.

Aggregation or a-part-of relationship:

It represents the situation where a class consists of several component classes. A class that is composed of other classes doesn't behave like its parts. It behaves very differently. The major properties of this relationship are transitivity and anti symmetry.

The questions whose answers will determine the distinction between the part and whole relationships are:

- Does the part class belong to the problem domain?
- Is the part class within the system's responsibilities?
- Does the part class capture more than a single value?(If not then simply include it as an attribute of the whole class)
- Does it provide a useful abstraction in dealing with the problem domain?
-

There are three types of aggregation relationships. They are:

Assembly:

It is constructed from its parts and an assembly-parts situation physically exists.

Container:

A physical whole encompasses but is not constructed from physical parts.

Collection member:

A conceptual whole encompasses parts that may be physical or conceptual. The container and collection are represented by hollow diamonds but composition is represented by solid diamond.

Use Case

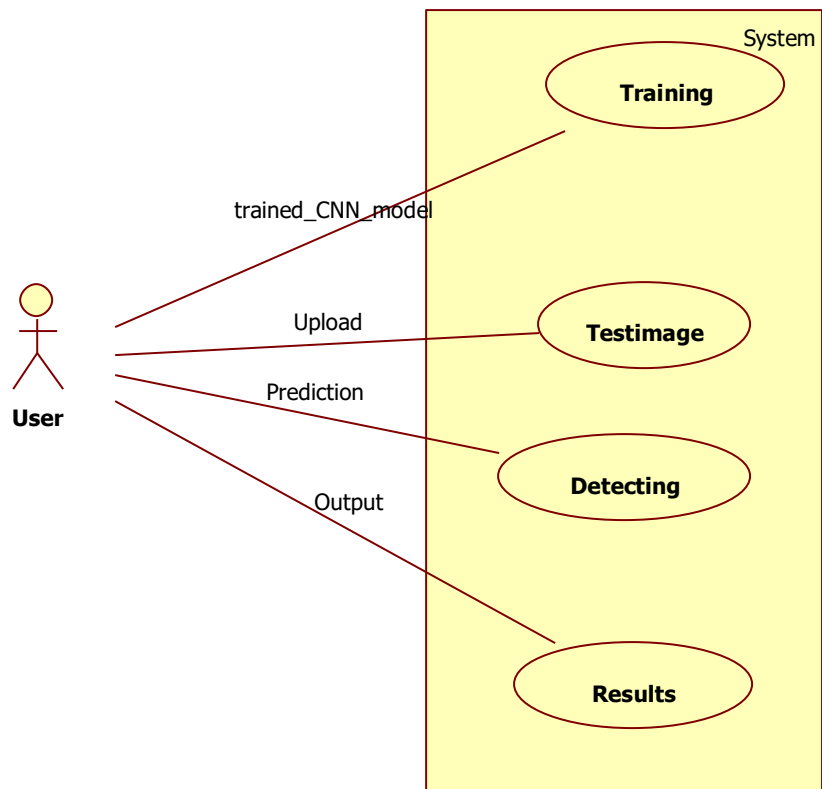


fig 3.3 usecase diagram of user

Sequence

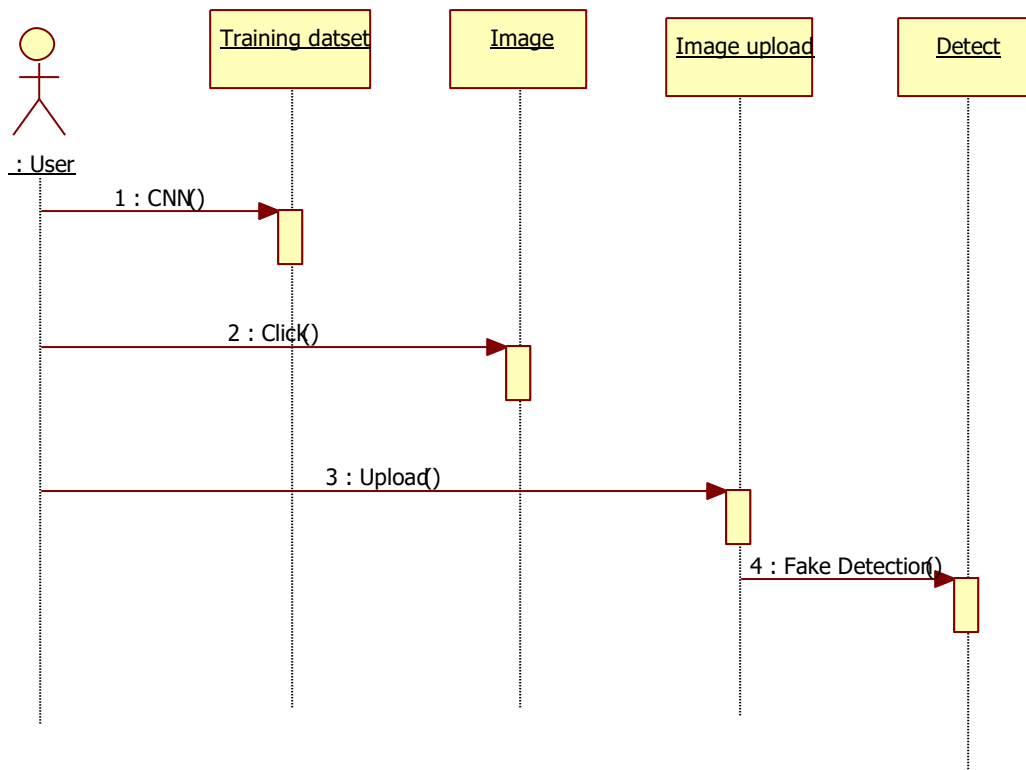
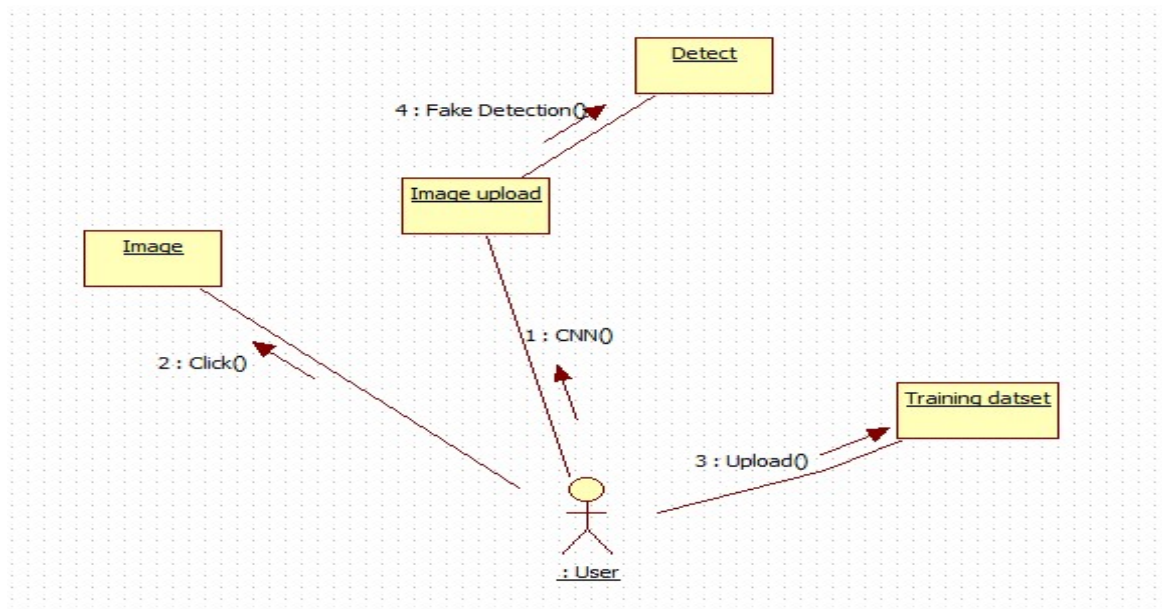


Fig 3.4 sequence diagram of user Colaboration



Activity

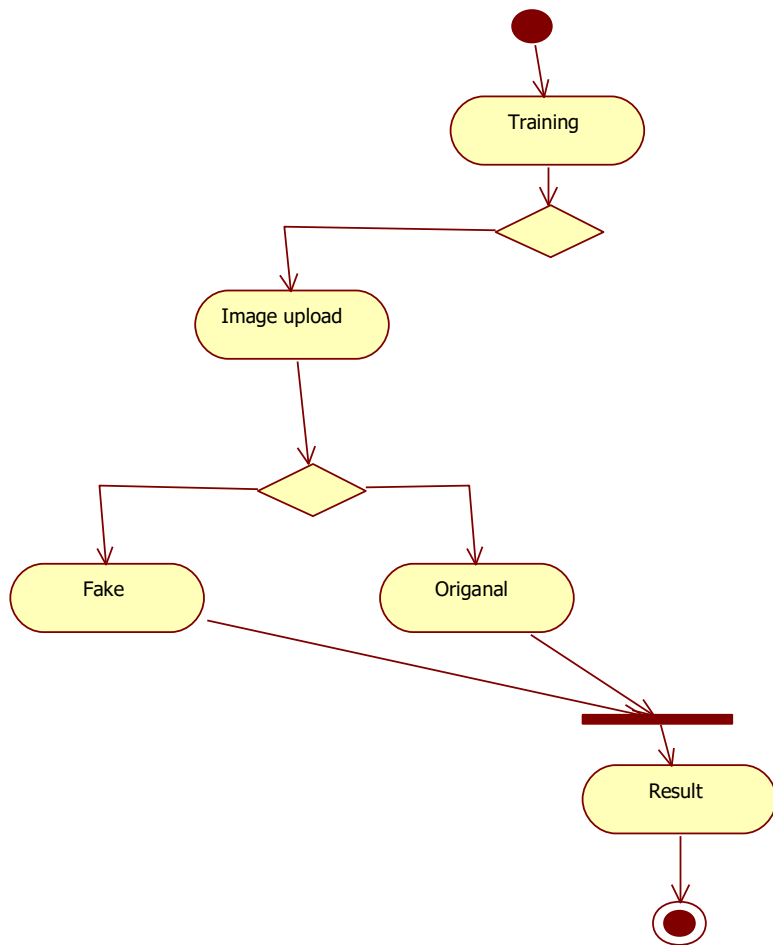


Fig 3.5 activity diagram of user Component

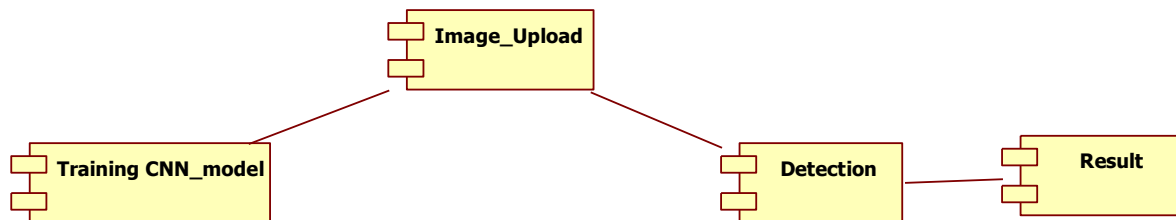


Fig 3.6 component diagram

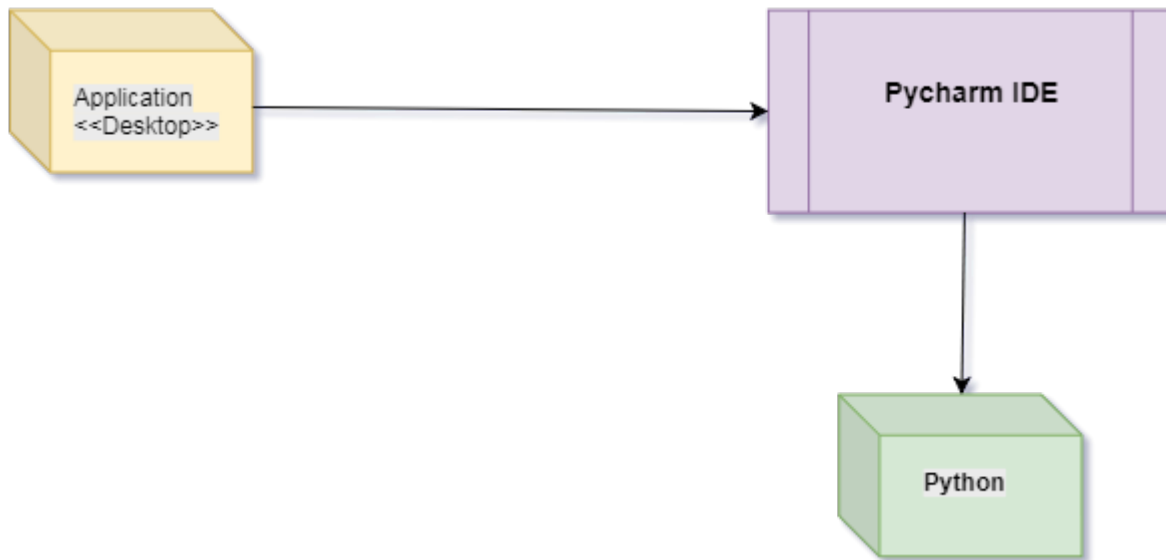


Fig 3.7 python deployment

CHAPTER 4-SYSTEM IMPLEMENTATION

4.1 PYTHON

Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands. Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, SmallTalk, and Unix shell and other scripting languages.

Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL).

Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

Input as CSV File

Reading data from CSV(comma separated values) is a fundamental necessity in Data Science. Often, we get data from various sources which can get exported to CSV format so that they can be used by other

systems. The Pandas library provides features using which we can read the CSV file in full as well as in parts for only a selected group of columns and rows.

The CSV file is a text file in which the values in the columns are separated by a comma. Let's consider the following data present in the file named input.csv. You can create this file using windows notepad by copying and pasting this data. Save the file as input.csv using the save As All files (*.*) option in notepad.

```
import pandas as pd
data= pd.read_csv('path/input.csv')
print(data)
```

Operations using NumPy:

NumPy is a Python package which stands for 'Numerical Python'. It is a library consisting of multidimensional array objects and a collection of routines for processing of array.

Using NumPy, a developer can perform the following operations –

- Mathematical and logical operations on arrays.
- Fourier transforms and routines for shape manipulation.
- Operations -related to linear algebra. NumPy has in-built functions for linear algebra and random number generation.

Key Features of Pandas

- Fast and efficient DataFrame object with default and customized indexing.
- Tools for loading data into in-memory data objects from different file formats.
- Data alignment and integrated handling of missing data.
- Reshaping and pivoting of date sets.
- Label-based slicing, indexing and subsetting of large data sets.
- Columns from a data structure can be deleted or inserted.
- Group by data for aggregation and transformations.
- High performance merging and joining of data.
- Time Series functionality.

4.2 SAMPLE CODE

AdminHome.py

```

#
from PyQt5 import QtCore, QtGui, QtWidgets
from CNN import build from Prediction
import Ui_Detection import sys class
Ui_AdminHome(object):
    def cnnbuild(self):
        try:
            build();
            self.showMessageBox("Message", "CNN build successfully.") except
Exception as e:
            print("Error=" + e.args[0])
tb = sys.exc_info()[2]
print(tb.tb_lineno)    def
detect(self):
    try:
        self.admn = QtWidgets.QDialog()
self.ui = Ui_Detection()
self.ui.setupUi(self.admn)
self.admn.show()    except Exception
as e:        print(e.args[0])        tb =
sys.exc_info()[2]
print(tb.tb_lineno)
    def showMessageBox(self, title, message):        msgBox =
QtWidgets.QMessageBox()
msgBox.setIcon(QtWidgets.QMessageBox.Information)

```



```

msgBox.setWindowTitle(title)      msgBox.setText(message)

msgBox.setStandardButtons(QtWidgets.QMessageBox.Ok)

    msgBox.exec_()

def setupUi(self, Dialog):

    Dialog.setObjectName("Dialog")

    Dialog.resize(700, 447)

    Dialog.setStyleSheet("background-color: rgb(170, 85, 0);")

self.label = QtWidgets.QLabel(Dialog)

self.label.setGeometry(QtCore.QRect(-70, 0, 841, 471))

self.label.setStyleSheet("image: url(../Detection/images/bg6.jpg);")

self.label.setObjectName("label")      self.pushButton =

QtWidgets.QPushButton(Dialog)

self.pushButton.setGeometry(QtCore.QRect(240, 100, 241, 51))

self.pushButton.setStyleSheet("background-color: rgb(170, 85, 0);\n"

"font: 12pt \"Franklin Gothic Heavy\";")

self.pushButton.setObjectName("pushButton")

self.pushButton.clicked.connect(self.cnnbuild)      self.pushButton_2 =

QtWidgets.QPushButton(Dialog)

self.pushButton_2.setGeometry(QtCore.QRect(240, 220, 241, 51))

self.pushButton_2.setStyleSheet("background-color: rgb(170, 85, 0);\n"

"font: 12pt \"Franklin Gothic Heavy\";")

self.pushButton_2.setObjectName("pushButton_2")

self.pushButton_2.clicked.connect(self.detect)

self.retranslateUi(Dialog)

    QtCore.QMetaObject.connectSlotsByName(Dialog)

def retranslateUi(self, Dialog):

```

```

    _translate = QtCore.QCoreApplication.translate

    Dialog.setWindowTitle(_translate("Dialog", "AdminHome"))

    self.label.setText(_translate("Dialog", ""))

self.pushButton.setText(_translate("Dialog", "Build CNN Model"))
self.pushButton_2.setText(_translate("Dialog", "Object Detection")) if
__name__ == "__main__":

    import sys

    app =
QtWidgets.QApplication(sys.argv)

    Dialog = QtWidgets.QDialog()
    ui = Ui_AdminHome()

    ui.setupUi(Dialog)

Dialog.show()

sys.exit(app.exec_())

```

CNN.py

```

from keras.models import Sequential from keras.layers
import Conv2D from keras.layers import MaxPooling2D
from keras.layers import Flatten from keras.layers import
Dense from keras.preprocessing.image import
ImageDataGenerator

def build():

    # Initialize the CNN

classifier = Sequential()

    # Convolution and Max pooling classifier.add(Conv2D(32, (3, 3),
input_shape=(128, 128, 3), activation='relu'))

classifier.add(MaxPooling2D(pool_size=(2, 2))) classifier.add(Conv2D(64, (3,
3), activation='relu')) classifier.add(MaxPooling2D(pool_size=(2, 2)))

```

```

classifier.add(Conv2D(128, (3, 3), activation='relu'))
classifier.add(MaxPooling2D(pool_size=(2, 2)))

# Flatten    classifier.add(Flatten()) # Full
connection   classifier.add(Dense(128,
activation='relu'))   classifier.add(Dense(5,
activation='softmax')) # Compile classifier
classifier.compile(optimizer='adam',
loss='categorical_crossentropy',
metrics=['accuracy'])

# Fitting CNN to the images

train_datagen = ImageDataGenerator(rescale=1. / 255, shear_range=0.2, zoom_range=0.2,
horizontal_flip=True) test_datagen = ImageDataGenerator(rescale=1. / 255)

training_set = train_datagen.flow_from_directory('./dataset/train', target_size=(128, 128),
batch_size=32,

                                class_mode='categorical')

test_set = test_datagen.flow_from_directory('./dataset/test', target_size=(128, 128), batch_size=32,

                                class_mode='categorical')

classifier.fit_generator(training_set, steps_per_epoch=25, epochs=50, validation_data=test_set,
validation_steps=200 / 32) # save model

classifier.save('cnn.h5') classifier.save_weights('weights_cnn.h5') Prediction.py

from PyQt5 import QtCore, QtGui, QtWidgets import os import sys from
Cnn_predict import predict class Ui_Detection(object): def browse_file(self):
fileName, _ = QtWidgets.QFileDialog.getOpenFileName(None, "Select Photo")

    print(fileName)

self.lineEdit.setText(fileName) def
detection_img(self):

```

```

try:
    image = self.lineEdit.text()    if
image == "" or image == "null":
    self.showMessageBox("Information", "Please Select Image")
else:
    result = predict(image)
    print("res=",result)
self.label_3.setText("Result : "+result) except
Exception as e:
    print(e.args[0])
tb = sys.exc_info()[2]
print(tb.tb_lineno) def
setupUi(self, Dialog):
    Dialog.setObjectName("Dialog")
    Dialog.resize(558, 409)
    Dialog.setStyleSheet("background-color: rgb(113, 75, 56);")
self.label = QtWidgets.QLabel(Dialog)
self.label.setGeometry(QtCore.QRect(190, 60, 301, 71))
self.label.setStyleSheet("color: rgb(255, 255, 255);\n"
"font: 16pt \"Georgia\";")    self.label.setObjectName("label")
self.label_2 = QtWidgets.QLabel(Dialog)
self.label_2.setGeometry(QtCore.QRect(110, 150, 101, 20))
self.label_2.setStyleSheet("color: rgb(255, 255, 255);\n"
"font: 12pt \"Georgia\";")
self.label_2.setObjectName("label_2")    self.lineEdit =
QtWidgets.QLineEdit(Dialog)

```

```

self.lineEdit.setGeometry(QtCore.QRect(110, 170, 291, 31))
self.lineEdit.setStyleSheet("font: 75 10pt \"Verdana\";")
self.lineEdit.setText("")
self.lineEdit.setObjectName("lineEdit")    self.pushButton =
QtWidgets.QPushButton(Dialog)
self.pushButton.setGeometry(QtCore.QRect(420, 170, 91, 31))
self.pushButton.setStyleSheet("color: rgb(255, 255, 255);\n"
"font: 10pt \"Georgia\";\n"
"background-color: rgb(57, 115, 172);")
self.pushButton.setObjectName("pushButton")
self.pushButton.clicked.connect(self.browse_file)    self.pushButton_3
= QtWidgets.QPushButton(Dialog)
self.pushButton_3.setGeometry(QtCore.QRect(190, 230, 121, 31))
self.pushButton_3.setStyleSheet("color: rgb(255, 255, 255);\n"
"font: 14pt \"Georgia\";\n"
"background-color: rgb(57, 115, 172);")
self.pushButton_3.setObjectName("pushButton_3")
self.pushButton_3.clicked.connect(self.detection_img)    self.label_3
= QtWidgets.QLabel(Dialog)
self.label_3.setGeometry(QtCore.QRect(120, 300, 411, 51))
self.label_3.setStyleSheet("color: rgb(255, 255, 255);\n"
"font: 16pt \"Georgia\";")
self.label_3.setObjectName("label_3")
self.retranslateUi(Dialog)

    QtCore.QMetaObject.connectSlotsByName(Dialog)
def retranslateUi(self, Dialog):

```

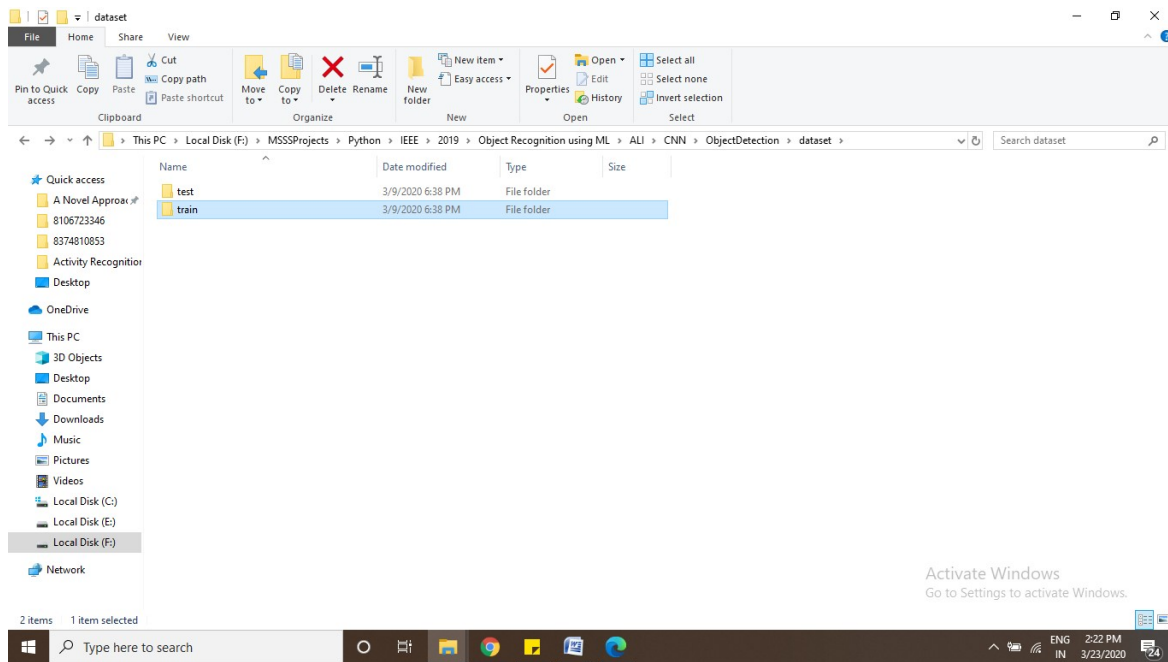
```

        _translate = QtCore.QCoreApplication.translate
Dialog.setWindowTitle(_translate("Dialog", "Image"))
self.label.setText(_translate("Dialog", "Fake Image Detection"))
self.label_2.setText(_translate("Dialog", "Select Image"))
self.pushButton.setText(_translate("Dialog", "Browse"))
self.pushButton_3.setText(_translate("Dialog", "Detect"))
self.label_3.setText(_translate("Dialog", "Result :"))

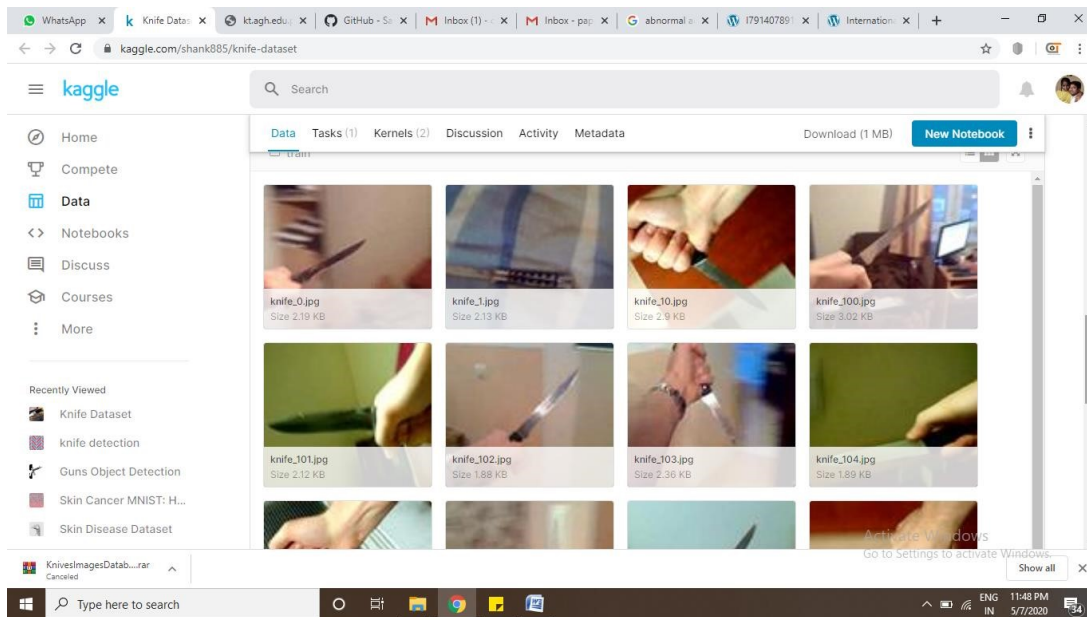
if __name__ == "__main__":
    import sys
    app =
QtWidgets.QApplication(sys.argv)
Dialog = QtWidgets.QDialog()
ui =
Ui_Dialog()
ui.setupUi(Dialog)
Dialog.show()
sys.exit(app.exec_())

```

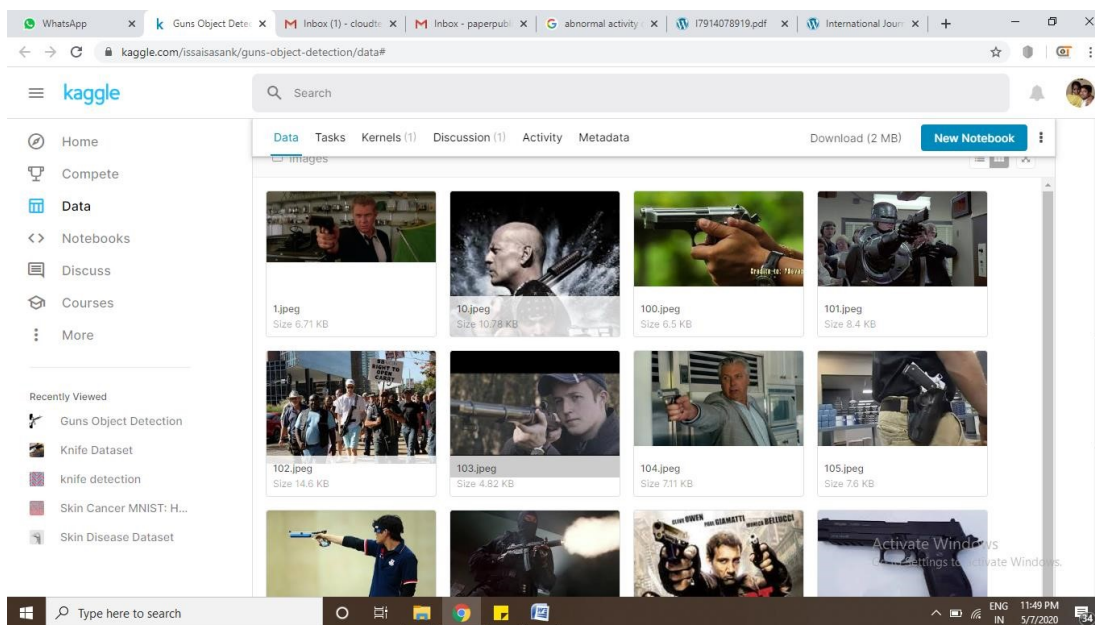
4.3 DATA SET



KNIFE



GUNS



4.4 SOFTWARE TESTING

Software testing is one of the main stages of project development life cycle to provide our cessation utilizer with information about the quality of the application and ours, in our Project we have under gone some stages of testing like unit testing where it's done in development stage of the project when we are in implementation of the application after the Project is yare we have done manual testing with different Case of all the different modules in the application we have even done browser compatibility testing in different web browsers in market, even we have done Client side validation testing on our application

Unit testing

The unit testing is done in the stage of implementation of the project only the error are solved in development stage some of the error we come across in development are given below

TESTING

Testing is the debugging program is one of the most critical aspects of the computer programming triggers, without programming that works, the system would never produce an output of which it was designed. Testing is best performed when user development is asked to assist in identifying all errors and bugs. The sample data are used for testing. It is not quantity but quality of the data used the matters of testing. Testing is aimed at ensuring that the system was accurately an efficiently before live operation commands.

Testing objectives

The main objective of testing is to uncover a host of errors, systematically and with minimum effort and time. Stating formally, we can say, testing is a process of executing a program with intent of finding an error.

A successful test is one that uncovers an as yet undiscovered error.

A good test case is one that has probability of finding an error, if it exists.

The test is inadequate to detect possibly present errors.

The software more or less confirms to the quality and reliable standards.

Levels of Testing

In order to uncover present in different phases we have the concept of levels of testing.

The basic levels of Testing:

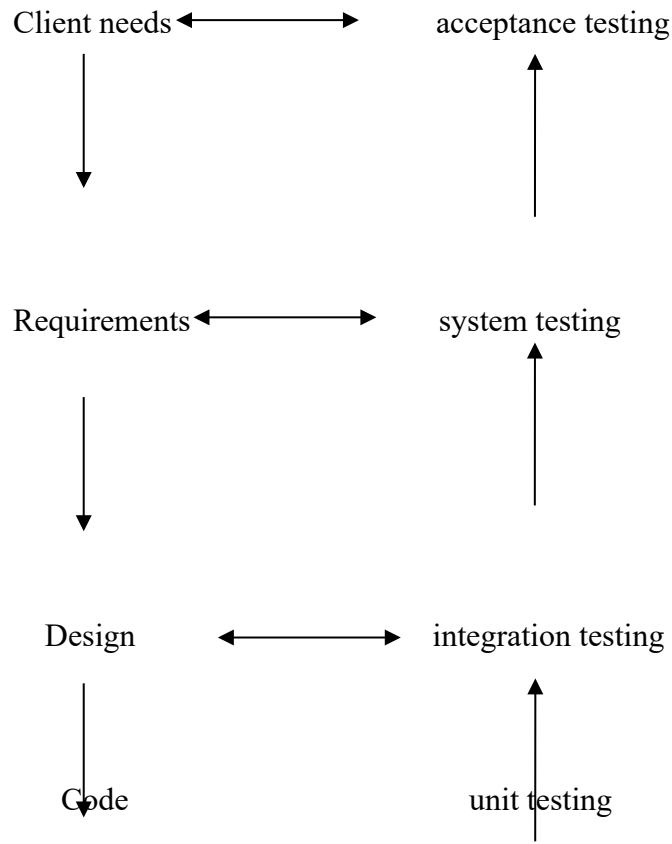


Figure: Levels of Testing

Code testing

This examines the logic of the program. For example, the logic for updating various sample data and with the sample files and directories were tested and verified.

Specification Testing

Executing this specification starting what the program should do and how it should performed under various conditions. Test cases for various situation and combination of conditions in all the modules are tested.

Unit testing:

In the unit testing we test each module individually and integrate with the overall system. Unit testing focuses verification efforts on the smallest unit of software design in the module. This is also known as module testing. The module of the system is tested separately. This testing is carried out during programming stage itself. In the testing step each module is found to work satisfactorily as regard to expected output from the module. There are some validation checks for fields also. For example the validation check is done for varying the user input given by the user which validity of the data entered. It is very easy to find error debut the system.

Each Module can be tested using the following two Strategies:

1. Black Box Testing
2. White Box Testing

BLACK BOX TESTING

Black box testing is a software testing technique in which functionality of the software under test (SUT) is tested without looking at the internal code structure, implementation details and knowledge of internal paths of the software. This type of testing is based entirely on the software requirements and specifications.

In Black Box Testing we just focus on inputs and output of the software system without bothering about internal knowledge of the software program.



The above Black Box can be any software system you want to test. For example : an operating system like Windows, a website like Google ,a database like Oracle or even your own custom application. Under Black Box Testing, you can test these applications by just focusing on the inputs and outputs without knowing their internal code implementation.

Black box testing - Steps

Here are the generic steps followed to carry out any type of Black Box Testing.

- Initially requirements and specifications of the system are examined.

- Tester chooses valid inputs (positive test scenario) to check whether SUT processes them correctly. Also some invalid inputs (negative test scenario) are chosen to verify that the SUT is able to detect them.
- Tester determines expected outputs for all those inputs.
- Software tester constructs test cases with the selected inputs.
- The test cases are executed.
- Software tester compares the actual outputs with the expected outputs.
- Defects if any are fixed and re-tested.

Types of Black Box Testing

There are many types of Black Box Testing but following are the prominent ones -

- **Functional testing** – This black box testing type is related to functional requirements of a system; it is done by software testers.
- **Non-functional testing** – This type of black box testing is not related to testing of a specific functionality, but non-functional requirements such as performance, scalability, usability.
- **Regression testing** – Regression testing is done after code fixes, upgrades or any other system maintenance to check the new code has not affected the existing code.

WHITE BOX TESTING

White Box Testing is the testing of a software solution's internal coding and infrastructure. It focuses primarily on strengthening security, the flow of inputs and outputs through the application, and improving design and usability. White box testing is also known as clear, open, structural, and glass box testing.

It is one of two parts of the "box testing" approach of software testing. Its counter-part, blackbox testing, involves testing from an external or end-user type perspective. On the other hand, Whitebox testing is based on the inner workings of an application and revolves around internal testing. The term "whitebox" was used because of the see-through box concept. The clear box or whitebox name symbolizes the ability to see through the software's outer shell (or "box") into its inner workings. Likewise, the "black box" in "black box testing" symbolizes not being able to see the inner workings of the software so that only the end-user experience can be tested

White box testing involves the testing of the software code for the following:

- Internal security holes
- Broken or poorly structured paths in the coding processes
- The flow of specific inputs through the code
- Expected output

- The functionality of conditional loops
- Testing of each statement, object and function on an individual basis

The testing can be done at system, integration and unit levels of software development. One of the basic goals of whitebox testing is to verify a working flow for an application. It involves testing a series of predefined inputs against expected or desired outputs so that when a specific input does not result in the expected output, you have encountered a bug.

We perform white box testing to give you a simplified explanation of white box testing, we have divided it into two basic steps. This is what testers do when testing an application using the white box testing technique:

STEP 1) UNDERSTAND THE SOURCE CODE

The first thing a tester will often do is learn and understand the source code of the application. Since white box testing involves the testing of the inner workings of an application, the tester must be very knowledgeable in the programming languages used in the applications they are testing. Also, the testing person must be highly aware of secure coding practices. Security is often one of the primary objectives of testing software. The tester should be able to find security issues and prevent attacks from hackers and naive users who might inject malicious code into the application either knowingly or unknowingly.

Step 2) CREATE TEST CASES AND EXECUTE

The second basic step to white box testing involves testing the application's source code for proper flow and structure. One way is by writing more code to test the application's source code. The tester will develop little tests for each process or series of processes in the application. This method requires that the tester must have intimate knowledge of the code and is often done by the developer. Other methods include manual testing, trial and error testing and the use of testing tools as we will explain further on in this article.

System testing: Once the individual module testing is completed, modules are assembled and integrated to perform as a system. The top down testing, which began from upper level to lower level module, was carried out to check whether the entire system is performing satisfactorily.

There are three main kinds of System testing:

- i. Alpha Testing
- ii. Beta Testing
- iii. Acceptance Testing

Alpha Testing:

This refers to the system testing that is carried out by the test team with the Organization.

Beta Testing:

This refers to the system testing that is performed by a selected group of friendly customers **Acceptance**

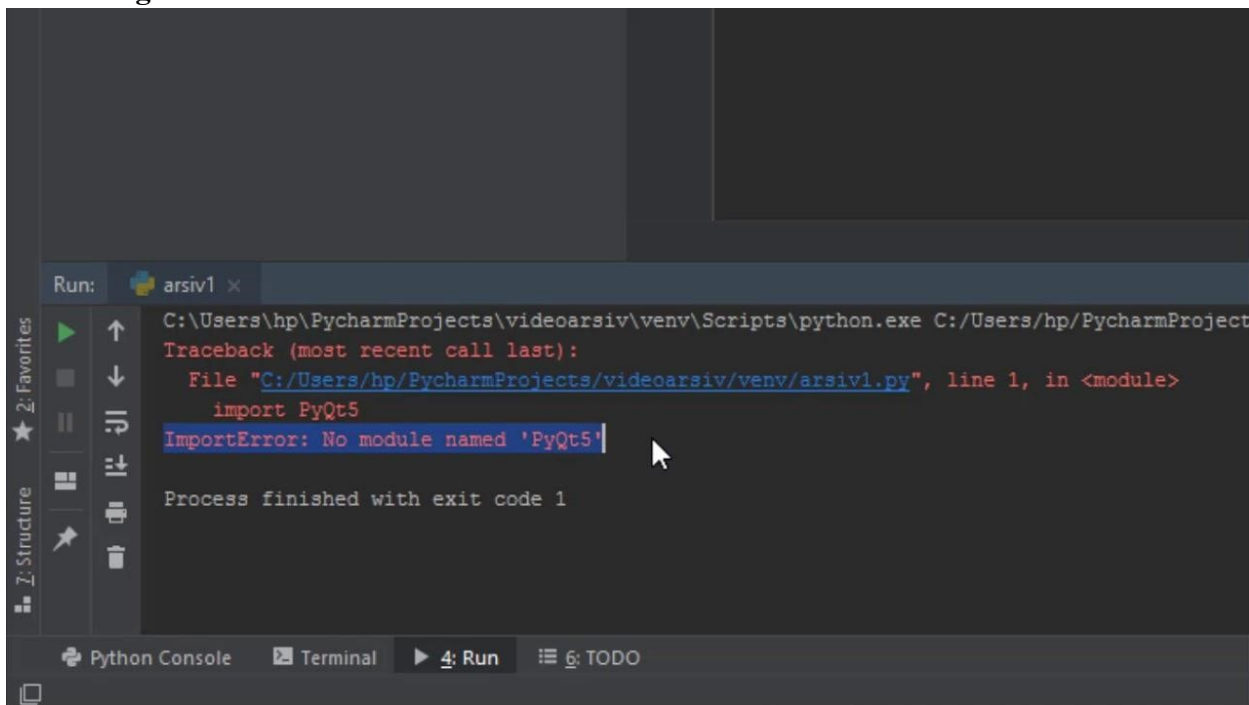
Testing:

This refers to the system testing that is performed by the customer to determine whether or not to accept the delivery of the system.

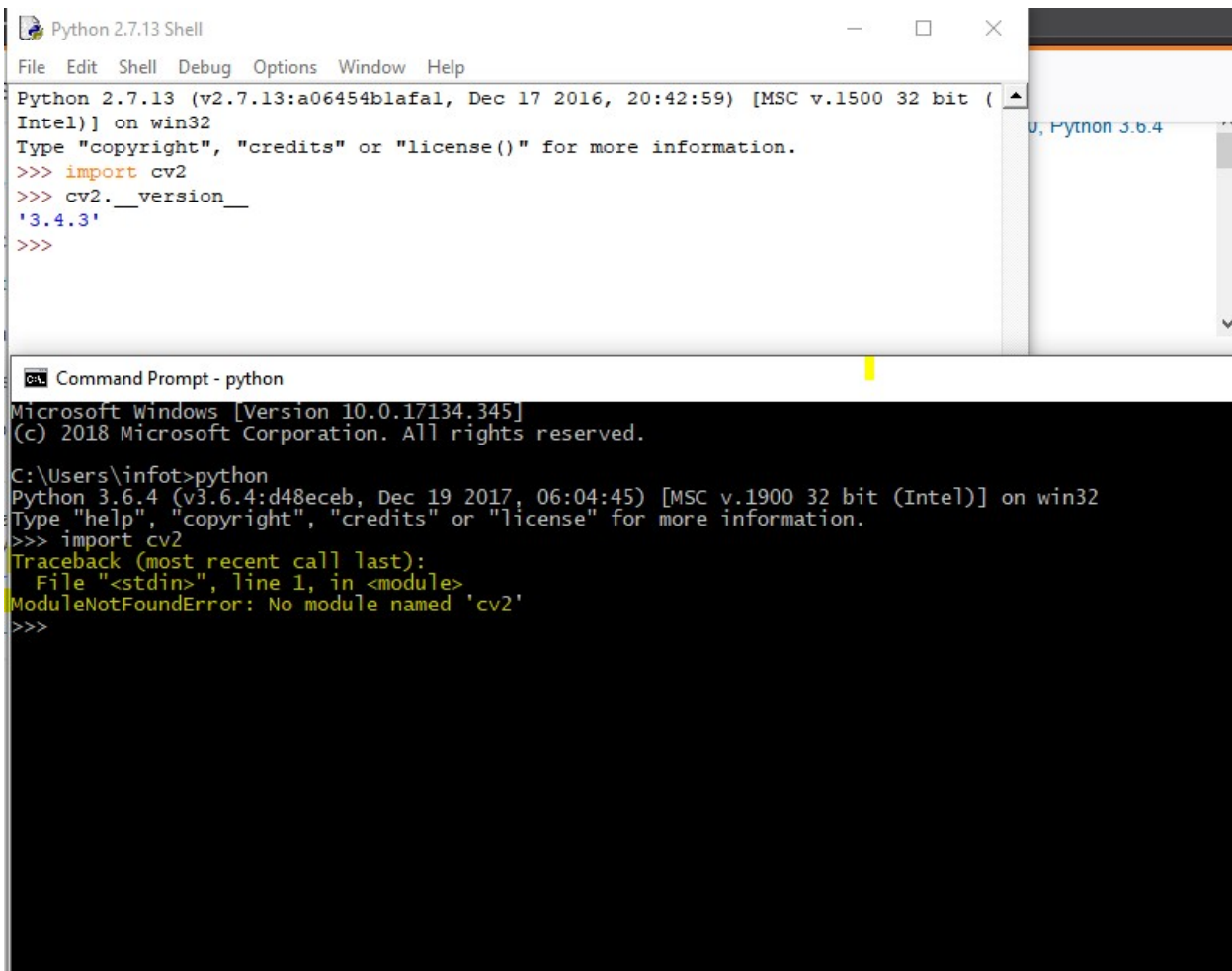
Integration Testing:

Data can be lost across an interface, one module can have an adverse effect on the other sub functions, when combined, may not produce the desired major functions. Integrated testing is the systematic testing for constructing the uncover errors within the interface. The testing was done with sample data. The developed system has run successfully for this sample data. The need for integrated test is to find the overall system performance.

Output testing: After performance of the validation testing, the next step is output testing. The output displayed or generated by the system under consideration is tested by asking the user about the format required by system.

Unit testing

When the pyqt5 module is not found the error will come as shown in fig



The top screenshot shows a Python 2.7.13 Shell window. The command prompt displays the Python version and architecture. The user enters the command `import cv2`, and the shell returns `cv2.__version__` as `'3.4.3'`.

The bottom screenshot shows a Command Prompt window running Python 3.6.4. The user enters the command `import cv2`, and the shell returns a `ModuleNotFoundError: No module named 'cv2'`.

When the cv2 module is not found the error will come as shown in fig

Manual testing on project application

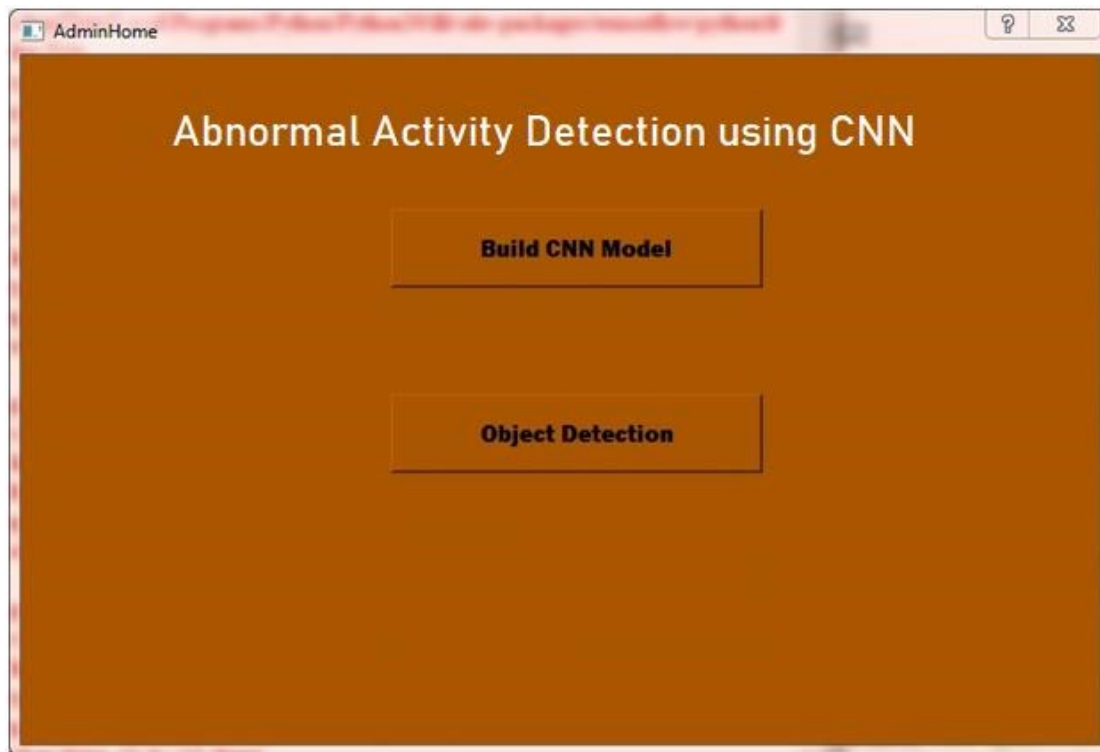
TEST CASES

SNO	IN PUT VALUE	TEST CASE	RESULT
1	User Name	NULL	YES
2	Password	NULL	YES
3	User Name & Password	Wrong	Validation Message

Tab.no-4.1 test cases

CHAPTER 5-RESULTS ANALYSIS, CONCLUSION AND FUTURE WORK

5.1 RESULTS



Home Screen

It can also be called the Index page. The system has two main features like:

- Build cnn model[4][6]
- Object detection

Building a CNN Model takes 3 hours time for getting detected. If we want to add new feature like fire[9][11], we have to store the features of fire[10] and use build CNN model option so that after sometime we can even detect the newly added feature. We have two different classifiers like abnormal and normal. We have two different datasets like train and test.

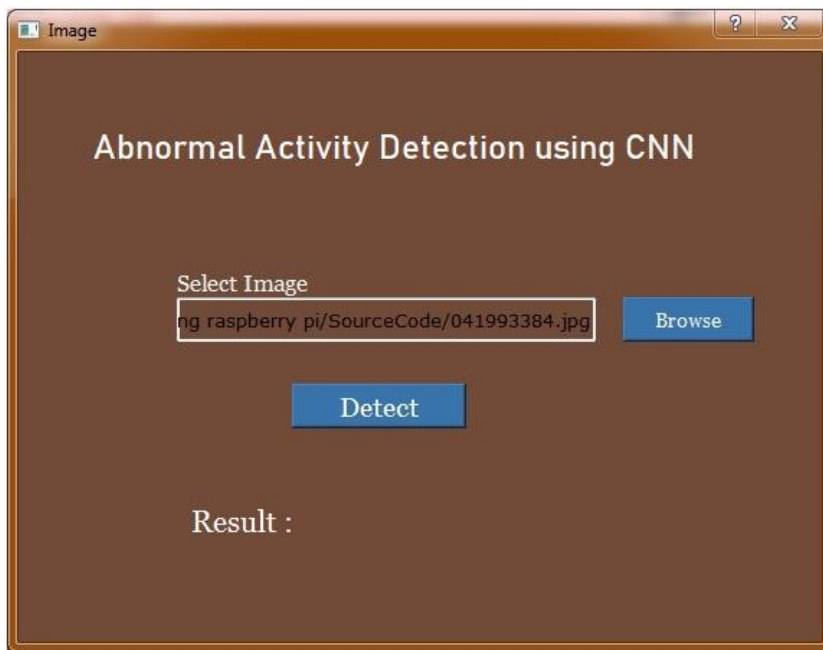
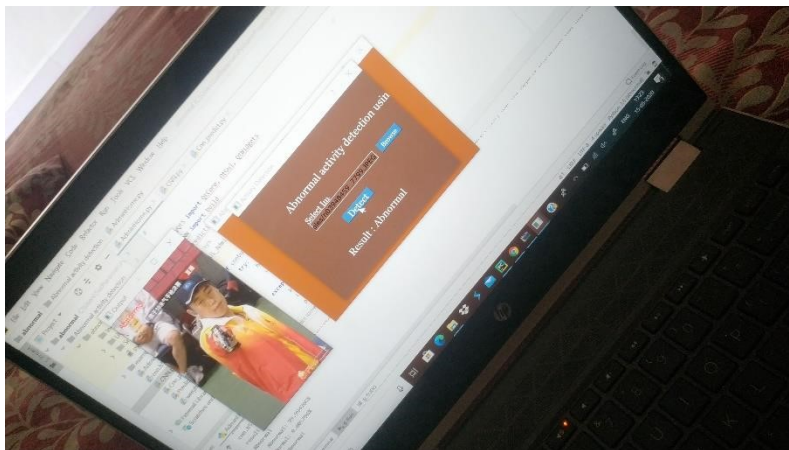


Image upload Screen

Each user once logged in, is a legitimate user of the system and therefore given the privilege to perform functions such as upload a video or image, browse file and view result. Here the user can upload the image and check the result whether it is normal or abnormal and we can even see the percentages of abnormality and normalities down.



5.2 CONCLUSION

A major challenge for automatic image analysis is that the sheer complexity of the visual task which has been mostly ignored by the current approaches. New technological breakthrough in the areas of digital computation and telecommunication has relevance for future applications of image processing¹. The satellite imaging and remote sensing applications programs of the future will feature a variety of sensors orbiting the earth. This technology is required for military and other types of surveillance, statistical data collection in the fields of forestry, agriculture, disaster prediction, weather prediction.

After researching the several methods that exists for human activity recognition it is noticed that the best suitable methods will be the ones that follow the unsupervised learning model, taking into consideration the kind of datasets available, In this paper, we presented a CNN model for the Abnormal Activities Detection (AAD) problem. We focused on a set of Abnormal activities extracted from a common exercise program for fall prevention, training our model data sampled from different classifiers , in order to explore the classification capabilities of each individual unit, as well as groups of units. Our experimental results indicate that convolutional models can be used to address the problem of activity recognition in the context of exercise programs. In this paper, we used Convolutional Neural Network model where the image or given input goes through five different layers. We can build our own CNN model and detect whether the image is normal or abnormal.

5.3 FUTURE WORK

The future of image processing will involve scanning the heavens for other intelligent life out in space. Also new intelligent, digital species created entirely by research scientists in various nations of the world will include advances in image processing applications. Due to advances in image processing and related technologies there will be millions and millions of robots in the world in a few decades time, transforming the way the world is managed. Advances in image processing and artificial intelligence will involve spoken commands, anticipating the information requirements of governments, translating languages, recognizing and tracking people and things, diagnosing medical conditions, performing surgery, reprogramming defects in human DNA, and automatic driving all forms of transport. With increasing power and sophistication of modern computing, the concept of computation can go beyond the present limits and in future, image processing technology will advance and the visual system of man can be replicated. The future trend in remote sensing will be towards improved sensors that record the same scene in many spectral channels. Graphics data is becoming increasingly important in image processing applications. The future image processing applications of satellite based imaging ranges from planetary exploration to surveillance applications. Using large scale homogeneous cellular arrays of simple circuits to perform image processing tasks and to demonstrate pattern-forming phenomena is an emerging topic. The convolutional neural network is an implementable alternative to fully connected neural networks and has evolved into a paradigm for future imaging techniques. The usefulness of this technique has applications in the areas of pattern formation, etc.

REFERENCES

- [1]. Fernando B, Gavves E, Oramas J, “Rank pooling for action recognition”, IEEE transactions on pattern analysis and machine intelligence, pp. 773-787,2019.
- [2]. Adrian Rosebrock , “Fire and smoke detection with Keras and Deep Learning” ,International Research Journal of Engineering and Technology, in November 18, 2019.
- [3] Andika Rachman “Convolutional Neural Network Architecture”: International Journal of Recent and Innovation Trends in Computing and Communication, may 17-2019.
- [4]. Pulkit Sharma, “Build an Image Classification Model using Convolutional Neural Networks” October 1, 2019.
- [5] .Sumit Saha “A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way” in Dec 15, 2018 .
- [6]. Eijaz Allibhai, “Building a Convolutional Neural Network (CNN) in Keras” in Oct 17, 2018.
- [7]. Alsheikh et al., ”A first approach to HAR based on deep learning models” June 2017, Vol.262,pp134-147.
- [8]. Hedde H W J, Bosman a.b. Giovanni I acca.a., Auturo Tejad a.c, Heinrich.J, Wortche.an Antoio Liotta.b, “Spatial anomaly detection using neighbourhood information”, Vol.17, pp. 41-56, 2017.
- [9]. Surbhi Narwani, “Real-Time Fire Detection for Video Surveillance Applications Using a Combination of Experts Based on Colour”, Shape and Motion, International Journal of Scientific and Research Publications, pp. 725-729, vol. 6, 2017.

- [10] . Shadab Dastgeer, Imranullah Khan, Shailendr K. Singh, “Fire Detection Using Image Processing Based on Color Analysis”, International Research Journal of Engineering and Technology, pp.27642769, vol. 3, 2017.
- [11] . Saumya Tiwari, Shuvabrata Bandopadhaya, “IoT Based Fire Alarm and Monitoring System”, International Journal of Innovations & Advancement in Computer Science, Vol. 6, 2017.
- [12]. Anagha V.Joshi, Nikita Hattiwale, “Optimal Fire Detection Using Image Processing”, International Journal of Recent and Innovation Trends in Computing and Communication, vol.5,pp.248-252,2017.
- [13]. Danqing Liu ,“A Practical Guide to ReLU” IEEE transactions on pattern analysis and machine intelligence, Nov 30, 2017.
- [14].SAGAR SHARMA “Activation Functions in Neural Networks Sigmoid, tanh, Softmax, ReLU, Leaky ReLU “ in Sep 6, 2017.
- [15]. Mattia Brusamento , “CNN for Short-Term Stocks Prediction using Tensorflow”, International Research Journal of Engineering and Technology, November 18, 2017.
- [16]. Bulling et.al, “Introduction to problems and highlighting capabilities classification models based on static and shallow features.”, pp.67-71, 2016.
- [17]. Daniel .B. Araya, Katarina Grolinger, “An deep learning framework using the Keras Sequential API in building cnn model”, Elsevier, Vol. 144, PP.191-206. 2016.
- [18] . Zeng et al., “Introduction to raw acceleration signals as input for a convolutional network”, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 1924-1932, 2016.
- [19] Hao-wei Yaoa, Ping Zhangb, “Introduction for Code for Design of Automatic Fire Alarm System”, pp.67-71, 2016.

- [20] ZenglinShi , YangdongYe, YunpengWu , “Rank-based pooling for deep convolutional neural networks” , International Journal of Scientific and Research Publications, 2016.