TRAFFIC SIGN DETECTION

A Project Report submitted
in partial fulfillment of the requirements
for the award of the degree of

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE & ENGINEERING

By

1. 18B01A0573 – K. V. Subbalakshmi
2. 18B01A0579 – N. Pavani
3. 18B01A0591 – K. Hiranmayee
4. 18B01A05A3 – M. Sita. Varshini
5. 19B05A0507 – G. M. Sailaja

Under the esteemed guidance of
Mrs. K. Ratna Kumari M. Tech
Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
SHRI VISHNU ENGINEERING COLLEGE FOR WOMEN(A)
(Approved by AICTE, Accredited by NBA & NAAC, Affiliated to JNTU Kakinada)
BHIMAVARAM – 534 202

2021 – 2022

SHRI VISHNU ENGINEERING COLLEGE FOR WOMEN(A)

(Approved by AICTE, Accredited by NBA & NAAC, Affiliated to JNTU Kakinada)
BHIMAVARAM – 534 202

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATE

*This is to certify that the project entitled "**Traffic Sign Detection**", is being submitted by **K. V. Subbalakshmi, N. Pavani, K. Hiranmayee, M. Sita. Varshini, G. M. Sailaja** bearing the **Regd. No. 18B01A0573, 18B01A0579, 18B01A0591, 18B01A05A3, 19B05A0507** in partial fulfillment of the requirements for the award of the degree of "**Bachelor of Technology** in **Computer Science & Engineering**" is a record of bonafide work carried out by her under my guidance and supervision during the academic year 2021–2022 and it has been found worthy of acceptance according to the requirements of the university.*

Internal Guide **Head of the Department**

External Examiner

ACKNOWLEDGMENT

The satisfaction that accompanies the successful completion of any task would be incomplete without the mention of the people who made it possible and whose constant encouragement and guidance has been a source of inspiration throughout the course of this project. I take this opportunity to express our gratitude to all those who have helped us in this project.

We wish to place our deep sense of gratitude to **Sri. K. V. Vishnu Raju, Chairman of SVES**, for his constant support on each and every progressive work of mine.

We wish to express our sincere thanks to **Dr. G. Srinivasa Rao, Principal of SVECW** for being a source of an inspirational and constant encouragement.

We wish to express our sincere thanks to **Dr. P. Srinivasa Raju, Vice-Principal of SVECW** for being a source of an inspirational and constant encouragement.

We wish to place our deep sense of gratitude to **Dr. P. Kiran Sree, Head of the** Department **of Computer Science & Engineering** for his valuable pieces of advice in completing this project successfully.

Our deep sense of gratitude and sincere thanks to **Mrs. K. Ratna Kumari, Assistant Professor** for her unflinching devotion and valuable suggestion throughout my project work.

Project Associates

1. **K. V. SUBBALAKSHMI - 18B01A0573**
2. **N. PAVANI - 18B01A0579**
3. **K. HIRANMAYEE - 18B01A0591**
4. **M. SITA VARSHINI - 18B01A05A3**
5. **G. M. SAILAJA - 19B05A0507**

ABSTRACT

Traffic signs are an integral part of our road infrastructure. They provide critical information, sometimes compelling recommendations, for road users including self-driving cars, which in turn requires them to adjust their driving behaviors to make sure they adhere to whatever road regulation currently enforced.

They keep traffic going by aiding travelers in reaching their destinations and providing them with advance notice of arrival, exit. There are several different types of traffic signs like speed limits, no entry, traffic signals, etc.

Vehicles are all set to drive along the road. As roads get denser with other vehicles, it is difficult for vehicles to identify all the traffic signs on the path. Chances of missing out on crucial signs that may lead to fatal accidents cannot be neglected. According to statistics, Driver behavior or error is identified as a major factor in 94 percent of crashes, and self-driving vehicles can help reduce driver errors.

To assist vehicles to overcome this problem we thought of implementing a machine learning model which can detect traffic signs.

CONTENTS

1. INTRODUCTION

With the rapid growth of technological development, vehicles have become an essential portion of in our routine lives. Because driving vehicles without follow traffic rules, it creates more and more intricate traffic on the road. As a result, it is one of the major reasons behind accidents every year.

In recent times road accidents are happening regularly in increasing manner across the world. Leading reason of most road accidents is the ignorance or unawareness of the traffic sign. The meaning of traffic sign is any entity, device, or board on the road that entity carries the rules, indicates the warning or provides other explanation regarding driving. Therefore, it also provides necessary information through traffic signals and traffic control devices to continue smooth car driving.

Traffic signs occupy an important position in the road traffic systems. The main function of the traffic signs is to display the contents that need to be noticed in the current road sections, to prompt the drivers in front of the road the danger and difficulty in the environment, to warn the driver to drive at the prescribed speed, to provide a favourable guarantee for safe driving.

Therefore, the detection and identification of traffic signs is a very important research direction, which is of great significance to prevent road traffic accidents and protect the personal safety of drivers.

Road traffic signs are divided in to two major categories of main signs and auxiliary signs. The main sign is divided into warning signs, prohibition signs, mandatory signs, guide signs, tourist signs and road construction and safety signs. Among them, prohibition signs mainly played a role in banning certain kinds of behaviour, a total of 43 categories.

Mandatory signs indicate the role of vehicles which is placed in the need to indicate vehicles, near the intersection. Warnings are mainly to alert drivers, vehicles pedestrians to beware of dangerous targets. They all play an important role in traffic signs.

Among them, the most common speed limit signs and the prohibition of left and right turning signs are of great significance for safe driving of drivers and therefore are the focus of the current research on traffic sign recognition.

The aim of this project is to develop a better system for the automatic detection and recognition of traffic signs with high accuracy and robustness under various complicated situations and disturbances.

Although the convolution neural network has made remarkable achievements, the current applications in the field of traffic sign detection and recognition are still not much. The reason for this is largely due to the lack of traffic sign data sets. Training and verifying a deep convolution neural network traffic sign recognition model requires a large amount of traffic sign data as a basis.

However, the open traffic sign datasets in India are relatively scarce compared with developing countries. More well-known traffic sign datasets now include GTSRB in Germany, GTSDB in Germany and KUL in Belgium.

In this project, GTSRB traffic sign dataset is used to traffic sign detection and recognition. This dataset includes many types of complex traffic signs such as sign tilt, uneven lighting, traffic sign with distraction, occlusion and similar background colors, as well as actual scene maps. Through a variety of complex and difficult to distinguish traffic signs to verify the ability of the algorithm.

Importance of Traffic Signs :-

Traffic signs are the salient speakers, preventing drivers and pedestrians from fatal risks on the road. More than 400 road accidents occur daily on Indian roads that cost lives and 3% of total annual GDP. Therefore, it is important to know 3 main categories of traffic signs in India to ensure road safety.

Below are the various types of traffic signs or symbols for Indian roads.

1. Mandatory Traffic Signs :-

2. Cautionary Traffic Signs :-

3. Informatory Traffic Signs :-

2. SYSTEM ANALYSIS

2.1 Existing System :-

- Output is displayed in text format which sometimes cannot be observed by the driver.

2.2 Proposed System :-

- Converts that text into speech format and makes the system able to read it out so that the driver can know the traffic sign even without watching the displayed text.

2.3 Feasibility Study :-

A feasibility study is an analysis that considers all of a project's relevant factors—including economic, technical, legal, and scheduling considerations—to ascertain the likelihood of completing the project successfully.

Whether a project is feasible or not can depend on several factors, including the project's cost and return on investment, meaning whether the project generated enough revenue or sales from consumers.

However, a feasibility study isn't only used for projects looking to measure and forecast financial gains. In other words, feasible can mean something different, depending on the industry and the project's goal.

Although feasibility studies can help project managers determine the risk and return of pursuing a plan of action, several steps and best practices should be considered before moving forward.

A feasibility study is an assessment of the practicality of a proposed plan or project. A feasibility study analyses the viability of a project to determine whether the project or venture is likely to succeed. The study is also designed to identify potential issues and problems that could arise from pursuing the project.

Technical Feasibility :-

A technical feasibility study can provide relevant context to the different aspects of your project and serve as a great planning tool by providing an overhead view of how your project can evolve during the course of its development, troubleshooting and tracking the progress of your project from concept to reality.

Technical feasibility deals with the existing technology, software and hardware requirements for the proposed system. The proposed system "TRAFFIC SIGN DETECTION" is planned to run on Python.

Economic Feasibility :-

Economic feasibility is a kind of cost-benefit analysis of the examined project, which assesses whether it is possible to implement it. This term means the assessment and analysis of a project's potential to support the decision-making process by objectively and rationally identifying its strengths, weaknesses, opportunities and risks associated with it, the resources that will be needed to implement the project, and an assessment of its chances of success.

This method is most frequently used for evaluating the effectiveness of a Python. It is also called as benefit analysis. In this project "TRAFFIC SIGN DETECTION" is developed on current equipment, existing software technology.

Behavioural Feasibility :-

It evaluates and estimates the user attitude or behaviour towards the development of new system.

It helps in determining if the system requires special effort to educate, retrain, transfer, and changes in employee's job status on new ways of conducting business.

This proposed system "TRAFFIC SIGN DETECTION" Application has much behavioural feasibility because users are provided with a better facility.

3. SYSTEM REQUIREMENTS SPECIFICATION

3.1 Software Requirements :-

- Tool : Spyder Notebook, Google Colab
- Programming Language : Python
- Operating System : Windows 10

3.2 Hardware Requirements :-

- RAM : 4GB
- Processor : Intel core i3 or Above.
- Hard Disk Space : 120 GB

3.3 Functional Requirements :-

These are the requirements that the end user specifically demands as basic facilities that the system should offer. All these functionalities need to be necessarily incorporated into the system as a part of the contract. These are represented or stated in the form of input to be given to the system, the operation performed and the output expected.

They are basically the requirements stated by the user which one can see directly in the final product, unlike the non-functional requirements.

Functional requirements are product features or functions those developers must implement to enable users to accomplish their tasks. So, it's important to make them clear both for the development team and the stakeholders. Generally, functional requirements describe system behaviour under specific conditions.

They are:

- Selected python as our analytics tool.
- Python includes many packages such as Pandas, NumPy, Matplotlib, etc.
- Machine Learning.

Non-functional Requirements :-

Nonfunctional requirements**,** not related to the system functionality, rather define *how* the system should perform.

These are basically the quality constraints that the system must satisfy according to the project contract. The priority or extent to which these factors are implemented varies from one project to other.

They are also called non-behavioural requirements. They basically deal with issues like:

- Portability - Portability, in relation to software, is a measure of how easily an application can be transferred from one computer environment to another.
- Maintainability – Maintainability is defined as the probability that a failed component or system will be restored or repaired to a specified condition within a specified period or time when maintenance is performed in accordance with prescribed procedures.
- Reliability - Reliability is defined as the probability that a product, system, or service will perform its intended function adequately for a specified period of time, or will operate in a defined environment without failure.
- Performance - Project performance management is the process of creating, implementing, and managing projects that contribute to the performance of an organization and its strategy.
- Reusability - Reusability is the quality of a code being used in different platforms for multiple functions.
- Flexibility - Project Flexibility refers to a project's ability to vary its output in a cost-effective manner within its lifecycle or a given time-frame.

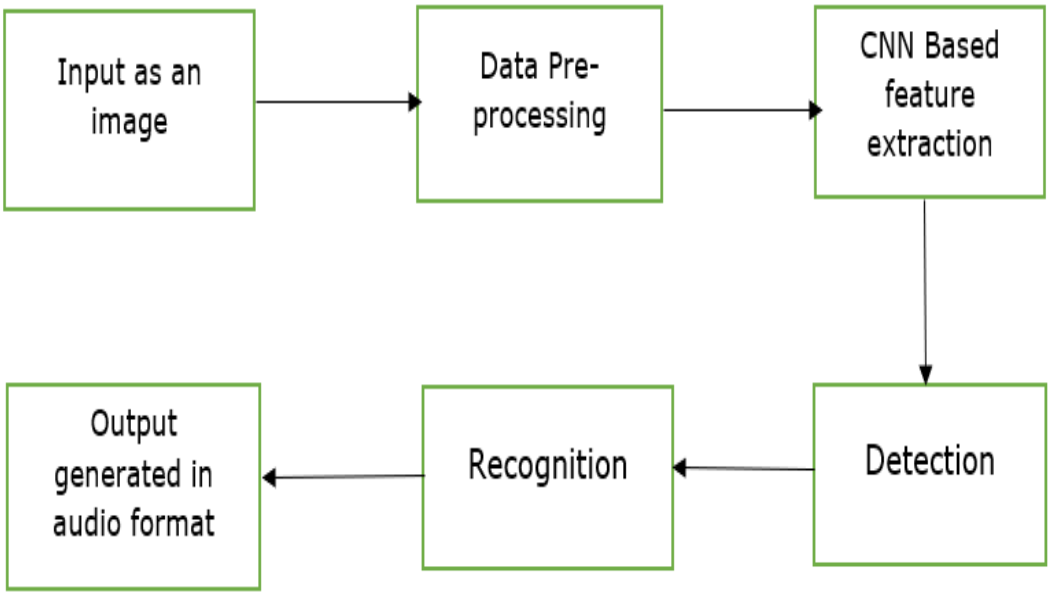
4. SYSTEM DESIGN

4.1 Introduction

System design is the process of designing the elements of a system such as the architecture, modules and components, the different interfaces of those components and the data that goes through that system. It is the process of defining, developing and designing systems which satisfies the specific needs and requirements of a business or organization.

A systemic approach is required for a coherent and well-running system. Bottom-Up or Top-Down approach is required to take into account all related variables of the system. A designer uses the modelling languages to express the information and knowledge in a structure of system that is defined by a consistent set of rules and definitions. The designs can be defined in graphical or textual modelling languages.

The purpose of the System Design process is to provide sufficient detailed data and information about the system and its system elements to enable the implementation consistent with architectural entities as defined in models and views of the system architecture.



Step – 1 :-The model takes an image as an input from user for specifying the traffic sign.

Step – 2 :-One of the limitations of the CNN model is that they cannot be trained on a different dimension of images. So, it is mandatory to have same dimension images in the dataset.

We'll check the dimension of all the images of the dataset so that we can process the images into having similar dimensions. In this dataset, the images have a very dynamic range of dimensions from 16*16*3 to 128*128*3 hence cannot be passed directly to the ConvNet model.

We need to compress or interpolate the images to a single dimension. Not, to compress much of the data and not to stretch the image too much we need to decide the dimension which is in between and keep the image data mostly accurate.

Step – 3 :- We build a CNN model to classify the images into their respective categories.

The architecture of our model is:

- 2 Conv2D layer (filter=32, kernel_size=(5,5), activation="relu")
- MaxPool2D layer (pool_size=(2,2))
- Dropout layer (rate=0.25)
- 2 Conv2D layer (filter=64, kernel_size=(3,3), activation="relu")
- MaxPool2D layer (pool_size=(2,2))
- Dropout layer (rate=0.25)
- Dense Fully connected layer (256 nodes, activation="relu")
- Dropout layer (rate=0.5)
- Dense layer (43 nodes, activation=" softmax")

Step – 4 :- Detects the traffic sign.

Step – 5 :- Recognises the traffic sign.

Step – 6 :- Predicted traffic sign label is given as output in both text and audio format.

System Analysis is the process that decomposes a system into its component pieces for the purpose of defining how well those components interact to accomplish the set requirements. The purpose of the System Design process is to provide sufficient detailed data and information about the system and its system elements to enable the implementation consistent with architectural entities as defined in models and views of the system architecture.

The purpose of the design phase is to plan a solution of the problem specified by the requirement document. This phase is the first step in moving from problem domain to the solution domain.

The design of a system is perhaps the most critical factor affecting the quality of the software, and has a major impact on the later phases, particularly testing and maintenance. The output of this phase is the design document. This document is similar to a blue print or plan for the solution, and is used later during implementation, testing and maintenance.

The design activity is often divided into two separate phase-system design and detailed design. System design, which is sometimes also called top-level design, aims to identify the modules that should be in the system, the specifications of these modules, and how they interact with each other to produce the desired results.

At the end of system design all the major data structures, file formats, output formats, as well as the major modules in the system and their specifications are decided.

A design methodology is a systematic approach to creating a design by application of set of techniques and guidelines. Most methodologies focus on system design. The two basic principles used in any design methodology are problem partitioning and abstraction. A large system cannot be handled as a whole, and so for design it’s partitioned into smaller systems.

Abstraction is a concept related to problem partitioning. When partitioning is used during design, the design activity focuses on one part of the system at a time. Since the part being designed interacts with other parts of the system, a clear understanding of the interaction is essential for property designing the part.

4.2 UML Diagrams

UML, short for Unified Modeling Language, is a standardized modeling language consisting of an integrated set of diagrams, developed to help system and software developers for specifying, visualizing, constructing, and documenting the artifacts of software systems, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems. The UML is a very important part of developing object oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects. Using the UML helps project teams communicate, explore potential designs, and validate the architectural design of the software.

As the strategic value of software increases for many companies, the industry looks for techniques to automate the production of software and to improve quality and reduce cost and time-to-market. These techniques include component technology, visual programming, patterns and frameworks.

Businesses also seek techniques to manage the complexity of systems as they increase in scope and scale. In particular, they recognize the need to solve recurring architectural problems, such as physical distribution, concurrency, replication, security, load balancing and fault tolerance.

Additionally, the development for the World Wide Web, while making some things simpler, has exacerbated these architectural problems. The Unified Modeling Language (UML) was designed to respond to these needs.

Some of them are :-

- Use Case Diagram
- Activity Diagram
- Sequence Diagram
- Collaboration Diagram
- Class Diagram

Use Case Diagram :-

In UML, use-case diagrams model the behavior of a system and help to capture the requirements of the system.

Use-case diagrams describe the high-level functions and scope of a system. These diagrams also identify the interactions between the system and its actors. The use cases and

actors in use-case diagrams describe what the system does and how the actors use it, but not how the system operates internally.

Use-case diagrams illustrate and define the context and requirements of either an entire system or the important parts of the system. You can model a complex system with a single use-case diagram, or create many use-case diagrams to model the components of the system.

You would typically develop use-case diagrams in the early phases of a project and refer to them throughout the development process.

Use-case diagrams are helpful in the following situations:

- Before starting a project, you can create use-case diagrams to model a business so that all participants in the project share an understanding of the workers, customers, and activities of the business.
- While gathering requirements, you can create use-case diagrams to capture the system requirements and to present to others what the system should do.
- During the analysis and design phases, you can use the use cases and actors from your use-case diagrams to identify the classes that the system requires.
- During the testing phase, you can use use-case diagrams to identify tests for the system.

The following topics describe model elements in use-case diagrams:

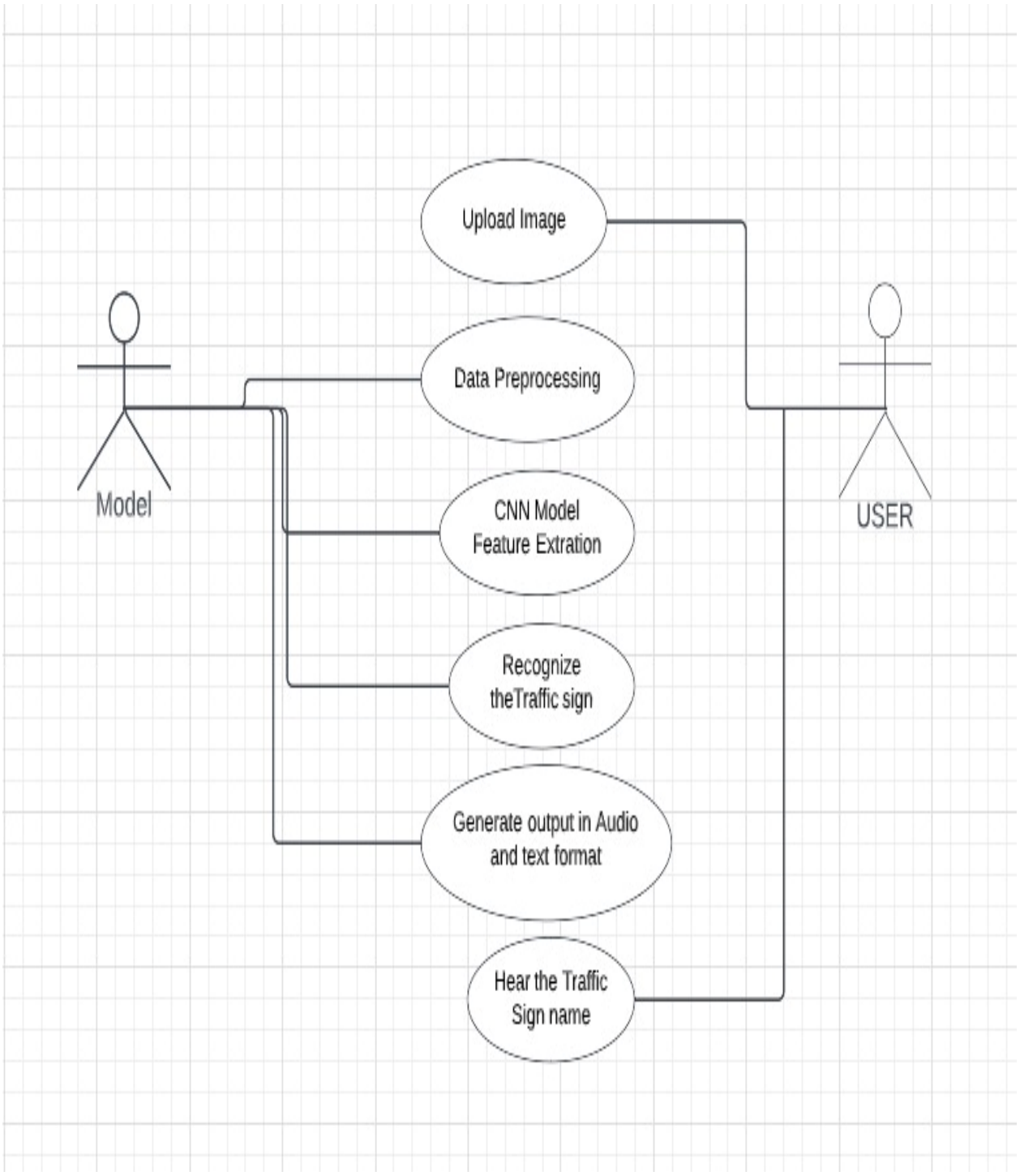
- **Use cases**
  A use case describes a function that a system performs to achieve the user's goal. A use case must yield an observable result that is of value to the user of the system.
- **Actors**
  An actor represents a role of a user that interacts with the system that you are modelling. The user can be a human user, an organization, a machine, or another external system.
- **Subsystems**
  In UML models, subsystems are a type of stereotyped component that represent independent, behavioural units in a system. Subsystems are used in class, component,

and use-case diagrams to represent large-scale components in the system that you are modelling.

- **Relationships in use-case diagrams**

  In UML, a relationship is a connection between model elements. A UML relationship is a type of model element that adds semantics to a model by defining the structure and behaviour between the model elements.

Use Case Diagram for Traffic Sign Detection :-



Activity Diagram :-

Activity diagram is another important behavioral diagram in UML diagram to describe dynamic aspects of the system. Activity diagram is essentially an advanced version of flow chart that modeling the flow from one activity to another activity.

Activity Diagrams describe how activities are coordinated to provide a service which can be at different levels of abstraction. Typically, an event needs to be achieved by some operations, particularly where the operation is intended to achieve a number of different things that require coordination, or how the events in a single use case relate to one another, in particular, use cases where activities may overlap and require coordination.

It is also suitable for modelling how a collection of use cases co-ordinate to represent business workflows

- Identify candidate use cases, through the examination of business workflows
- Identify pre- and post-conditions (the context) for use cases
- Model workflows between/within use cases
- Model complex workflows in operations on objects
- Model in detail complex activities in a high level activity Diagram

Basic Activity Diagram Notations and Symbols

Initial State or Start Point :-

- A small filled circle followed by an arrow represents the initial action state or the start point for any activity diagram. For activity diagram using swim lanes, make sure the start point is placed in the top left corner of the first column.

Activity or Action State :-

- An action state represents the non-interruptible action of objects. You can draw an action state in Smart Draw using a rectangle with rounded corners.

Action Flow :-

- Action flows, also called edges and paths, illustrate the transitions from one action state to another. They are usually drawn with an arrowed line.

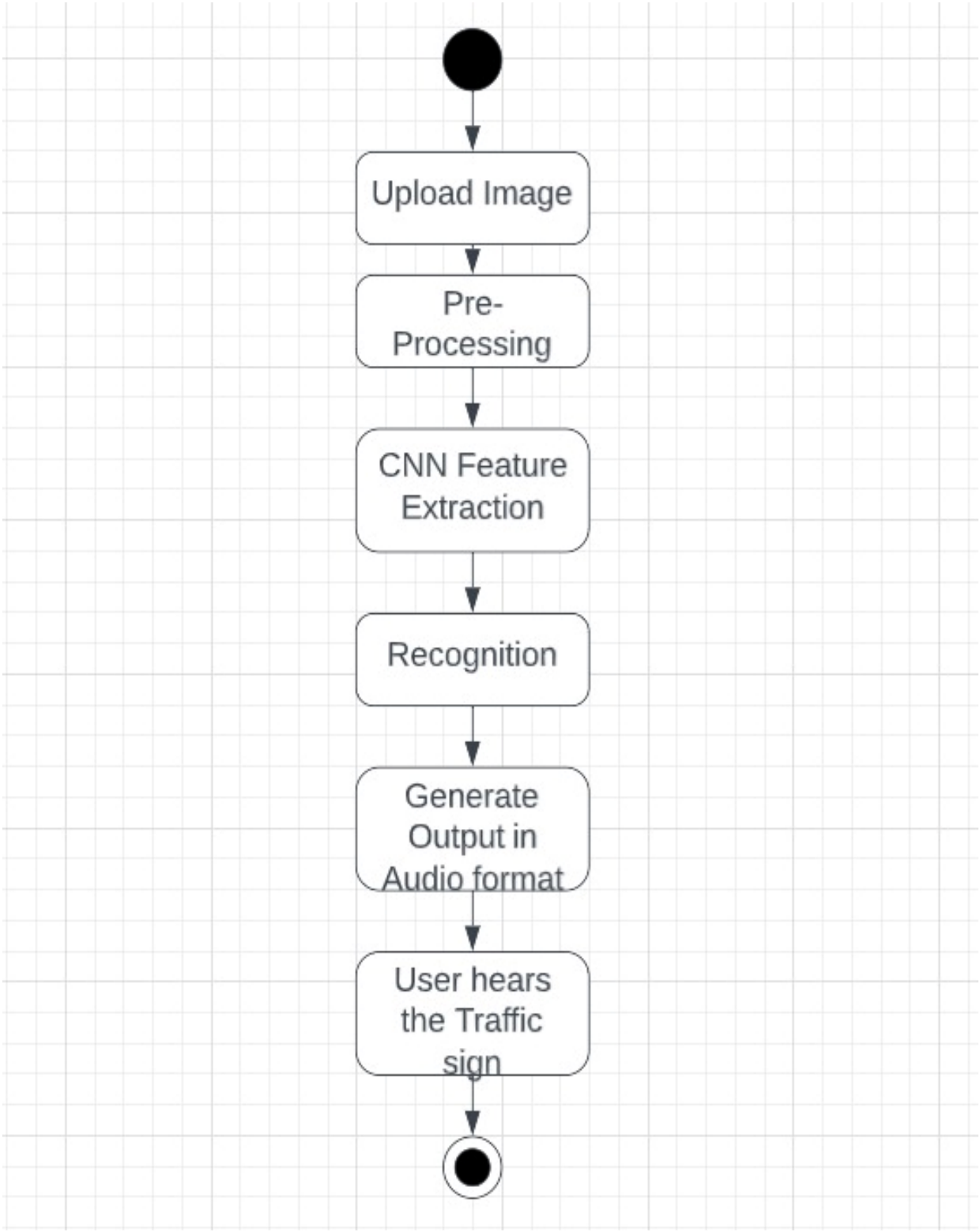
Object Flow :-

- Object flow refers to the creation and modification of objects by activities. An object flow arrow from an action to an object means that the action creates or influences the object. An object flow arrow from an object to an action indicates that the action state uses the object.

Decisions and Branching :-

- A diamond represents a decision with alternate paths. When an activity requires a decision prior to moving on to the next activity, add a diamond between the two activities. The outgoing alternates should be labelled with a condition or guard expression. You can also label one of the paths "else."

Activity Diagram for Traffic Sign Detection :-



Sequence Diagram :-

A sequence diagram or system sequence diagram (SSD) shows object interactions arranged in time sequence in the field of software engineering. It depicts the objects involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of scenario. Sequence diagrams are typically associated with use case realizations in the logical view of the system under development. Sequence diagrams are sometimes called event diagrams or event scenarios.

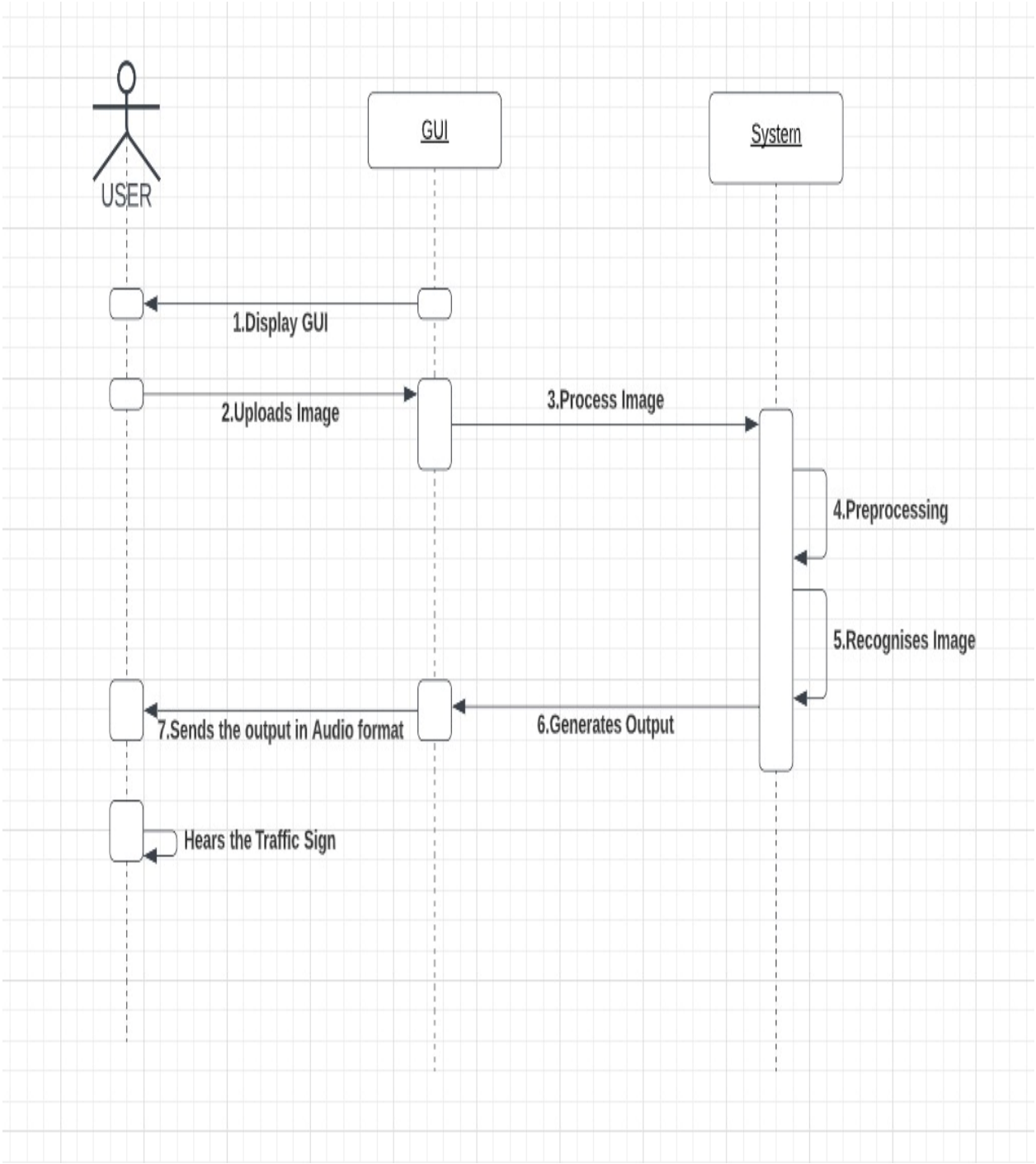
For a particular scenario of a use case, the diagrams show the events that external actors generate, their order, and possible inter-system events.[1] All systems are treated as a black box; the diagram places emphasis on events that cross the system boundary from actors to systems. A system sequence diagram should be done for the main success scenario of the use case, and frequent or complex alternative scenarios.

Benefits of sequence diagrams :-

Sequence diagrams can be useful references for businesses and other organizations. Try drawing a sequence diagram to:

- Represent the details of a UML use case.
- Model the logic of a sophisticated procedure, function, or operation.
- See how objects and components interact with each other to complete a process.
- Plan and understand the detailed functionality of an existing or future scenario.

Sequence Diagram for Traffic Sign Detection :-



Collaboration Diagram :-

The collaboration diagram is used to show the relationship between the objects in a system. Both the sequence and the collaboration diagrams represent the same information but differently. Instead of showing the flow of messages, it depicts the architecture of the object residing in the system as it is based on object-oriented programming. An object consists of several features. Multiple objects present in the system are connected to each other. The collaboration diagram, which is also known as a communication diagram, is used to portray the object's architecture in the system.

Following are the components of a component diagram that are enlisted below:

Objects :-

The representation of an object is done by an object symbol with its name and class underlined, separated by a colon. In the collaboration diagram, objects are utilized in the following ways:

- The object is represented by specifying their name and class.
- It is not mandatory for every class to appear.
- A class may constitute more than one object.
- In the collaboration diagram, firstly, the object is created, and then its class is specified.
- To differentiate one object from another object, it is necessary to name them.

Actors :-

In the collaboration diagram, the actor plays the main role as it invokes the interaction. Each actor has its respective role and name. In this, one actor initiates the use case.

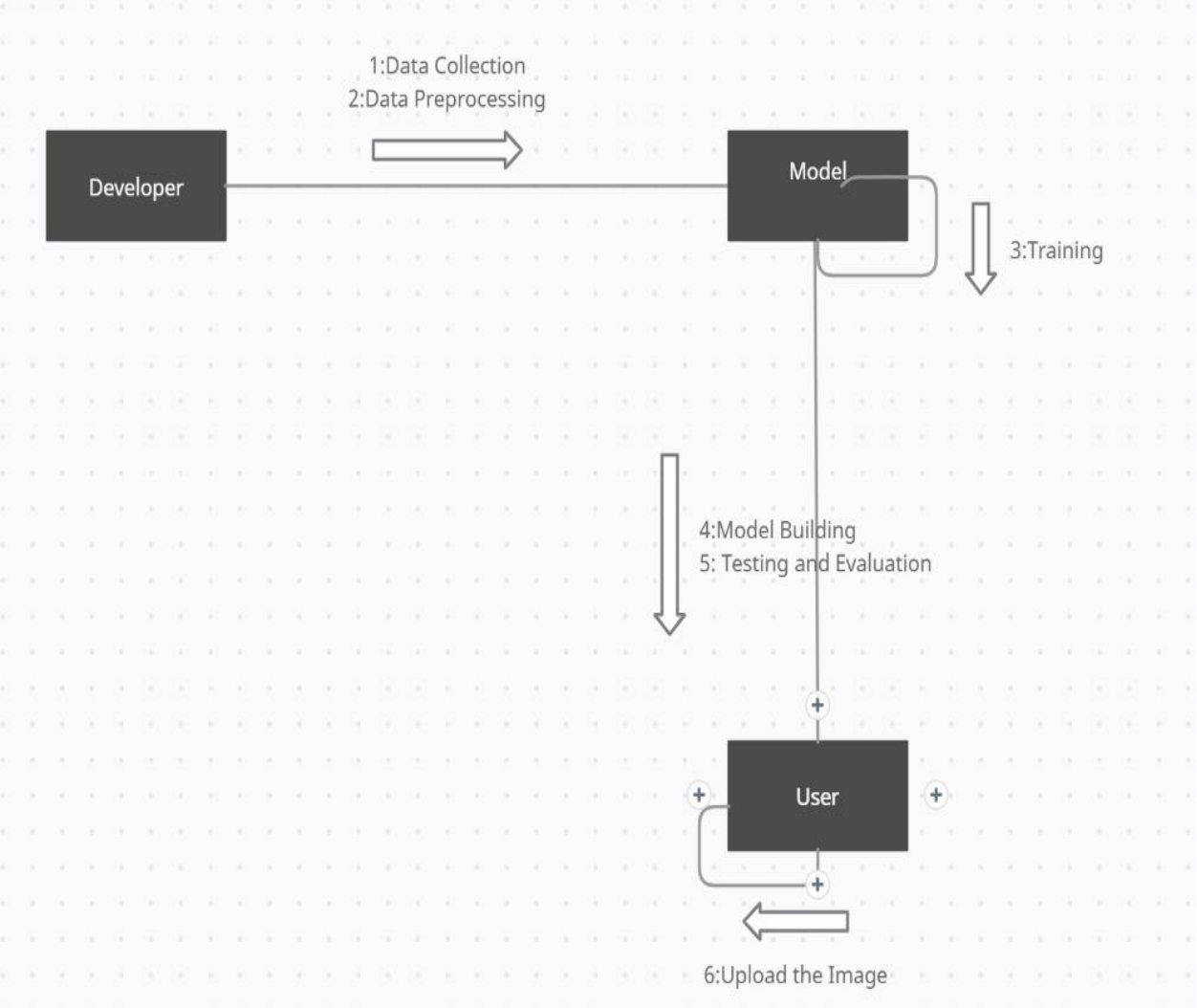
Links :-

The link is an instance of association, which associates the objects and actors. It portrays a relationship between the objects through which the messages are sent. It is represented by a solid line. The link helps an object to connect with or navigate to another object, such that the message flows are attached to links.

Messages :-

It is a communication between objects which carries information and includes a sequence number, so that the activity may take place. It is represented by a labelled arrow, which is placed near a link. The messages are sent from the sender to the receiver, and the direction must be navigable in that particular direction. The receiver must understand the message.

Collaboration Diagram for Traffic Sign Detection :-



Class Diagram :-

The UML Class diagram is a graphical notation used to construct and visualize object oriented systems. A class diagram in the Unified Modelling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's:

- classes,
- their attributes,
- operations (or methods),
- and the relationships among objects.

A Class is a blueprint for an object. Objects and classes go hand in hand. We can't talk about one without talking about the other. And the entire point of Object-Oriented Design is not about objects, it's about classes, because we use classes to create objects. So, a class describes what an object will be, but it isn't the object itself.

In fact, classes describe the type of objects, while objects are usable instances of classes. Each Object was built from the same set of blueprints and therefore contains the same components (properties and methods). The standard meaning is that an object is an instance of a class and object - Objects have states and behaviours.

UML Class Notation

A class represent a concept which encapsulates state (attributes) and behaviour (operations). Each attribute has a type. Each operation has a signature. The class name is the only mandatory information.

Class Name:

- The name of the class appears in the first partition.

Class Attributes:

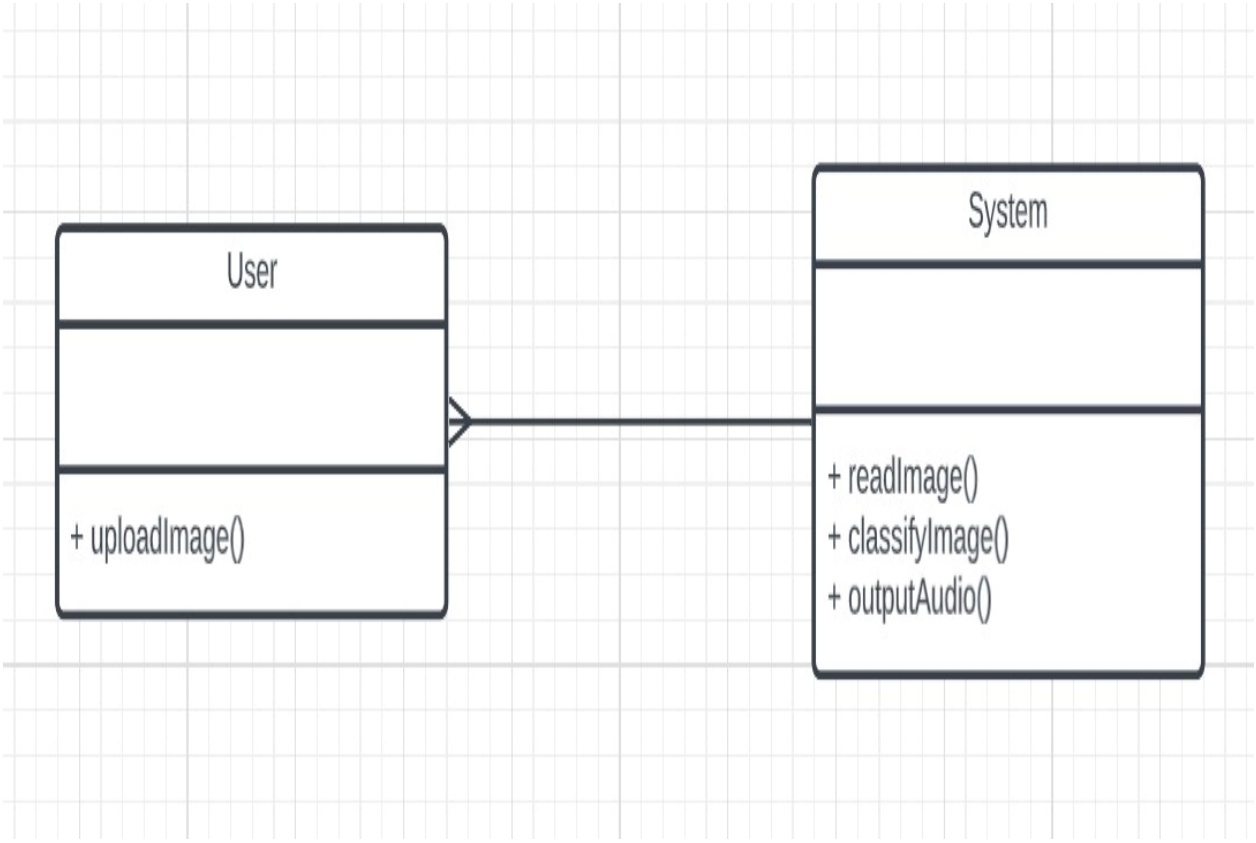
- Attributes are shown in the second partition.
- The attribute type is shown after the colon.
- Attributes map onto member variables (data members) in code.

Class Operations (Methods):

- Operations are shown in the third partition. They are services the class provides.

- The return type of a method is shown after the colon at the end of the method signature.
- The return type of method parameters is shown after the colon following the parameter name. Operations map onto class methods in code.

Class diagram for Traffic Sign Detection :-



5. SYSTEM IMPLEMENTION

5.1 Introduction

Classification with artificial neural networks is a very popular approach to solve pattern recognition problems. A neural network is a mathematical model based on connected via each other neural units – artificial neurons – similarly to biological neural networks. Typically, neurons are organized in layers, and the connections are established between neurons from only adjacent layers. The input low-level feature vector is put into first layer and, moving from layer to layer, is transformed to the high-level features vector. The output layer neurons amount is equal to the number of classifying classes. Thus, the output vector is the vector of probabilities showing the possibility that the input vector belongs to a corresponding class.

Today, classifying with convolutional neural networks is the state of the art pattern recognition method in computer vision. Unlike traditional neural networks, which works with one-dimensional feature vectors, a convolutional neural network takes a two-dimensional image and consequentially processes it with convolutional layers.

Each convolutional layer consists of a set of trainable filters and computes dot productions between these filters and layer input to obtain an activation map. These filters are also known as kernels and allow detecting the same features in different locations.

The CNN learns the filters' values during the model training process. Some parameters are necessitated to specify here like the number of filters, network architecture being used, size of filter and many other, prior to the training process. Recognition of unseen patterns from image becomes more efficient when number of filters are increased in number. Also, the extraction level and quality of pattern evaluation gets improved to a significant level. Some of the important model parameters are:

Filters :-

The convolution operation on an image is used to identify dependencies of the image. A feature map is produced as a result when any filter slides over an input image [18]. Different feature maps are generated in turn when convolution operation is performed with another filter. The filters and image are stored for operation as numeric matrices.

Epochs :-

It is an entity used to signify how many times the entire dataset is passed forward and backward through the neural network. Although the dataset is too big in size to feed to the system even once, but it is required to make multiple submissions to get multiple tests generated optimal results.

Batch size :-

Different from number of batches, it is a number which represents the number of training examples in a batch. As the whole dataset can't be passed at once to the neural network model, batch numbers are nothing but the division of a dataset so that it is easy to process with accuracy.

Learning rate :-

There is an iterative optimization algorithm in machine learning called as gradient descent. This iterative algorithm works multiple times to get optimal results. A parameter here in is known as learning rate. This also helps in making an under-fit graph to fit optimally.

Loss regularization :-

It refers to the changes or transformations that are made to reduce the generalization error, and not the training error. These modifications are done on the learning algorithm. It is an important factor to prevent over-fitting and maintain precision along with accuracy. Some of the methods used to perform data regularization include using a dropout layer, introducing weight penalty, augmentation on the dataset and incorporating early stopping for effectively tuning hyper-parameters like epochs, number of batches etc.

Dropout rate :-

Some nodes in neural networks are randomly chosen by the dropout layer for their movement along their incoming and outgoing connections. This layer operates on hidden layer as well as the input layer. It can also be said as an ensembling technique, and the training neural network dropout can be seen as an ensemble collection of 2^n thinned networks. It is useful for different random subsets of neurons which help in learning more robust features.

5.2 Project Modules

1. Data Pre-processing
2. Model Building
3. Model Training
4. Testing and Validation
5. Model Deployment

1. Data Pre-processing :-

Data pre-processing is a process of preparing the raw data and making it suitable for a machine learning model. It is the first and crucial step while creating a machine learning model.

When creating a machine learning project, it is not always a case that we come across the clean and formatted data. And while doing any operation with data, it is mandatory to clean it and put in a formatted way. So, for this we use data pre-processing task.

A real-world data generally contains noises, missing values, and maybe in an unusable format which cannot be directly used for machine learning models. Data preprocessing is required tasks for cleaning the data and making it suitable for a machine learning model which also increases the accuracy and efficiency of a machine learning model.

2. Model Building :-

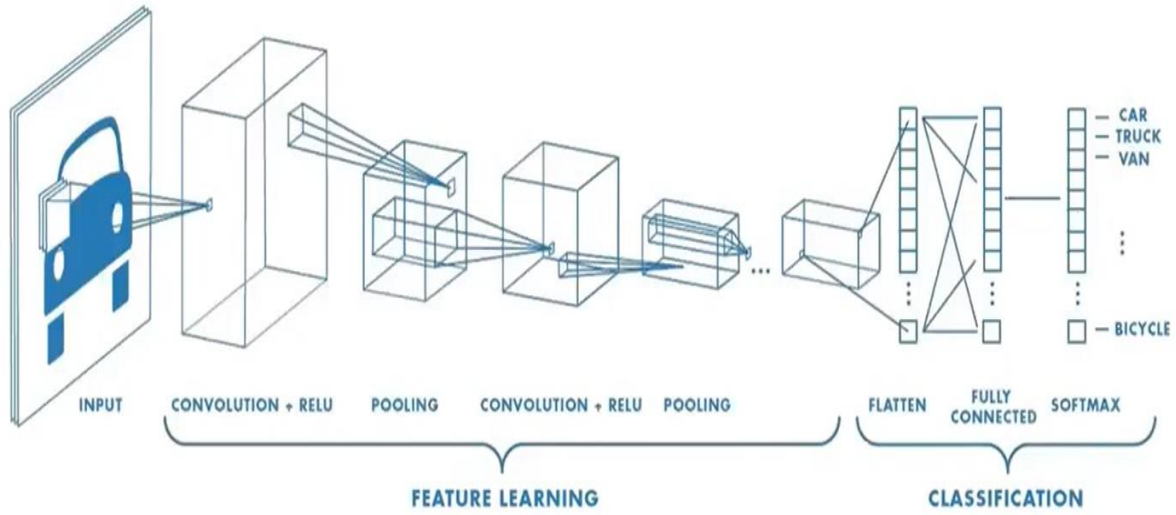
This is a very crucial step in our deep learning model building process. We have to define how our model will look and that requires

- Importing the libraries
- Initializing the model
- Adding CNN (Convolution Neural Network) Layers
- Adding Dense layers

A great way to use deep learning to classify images is to build a convolutional neural network (CNN).The Keras library in Python makes it pretty simple to build a CNN. Computers see images using pixels. Pixels in images are usually related. For example, a certain group of

pixels may signify an edge in an image or some other pattern. Convolutions use this to help identify images.

A convolution multiplies a matrix of pixels with a filter matrix or 'kernel' and sums up the multiplication values. Then the convolution slides over to the next pixel and repeats the same process until all the image pixels have been covered.



3. Model Training :-

Model training in machine language is the process of feeding an ML algorithm with data to help identify and learn good values for all attributes involved. There are several types of machine learning models, of which the most common ones are supervised and unsupervised learning.

Supervised learning is possible when the training data contains both the input and output values. Each set of data that has the inputs and the expected output is called a supervisory signal. The training is done based on the deviation of the processed result from the documented result when the inputs are fed into the model.

Unsupervised learning involves determining patterns in the data. Additional data is then used to fit patterns or clusters. This is also an iterative process that improves the accuracy based on the correlation to the expected patterns or clusters. There is no reference output dataset in this method.

4. Testing and Validation :-

Machine learning model evaluation focuses on the overall performance of the model. Such evaluations can consist of performance metrics and curves, and perhaps examples of incorrect predictions.

This way of model evaluation is a great way to monitor your model's outcome between different versions. However, it does not tell us a lot about the reasons behind the failures and the specific model behaviors.

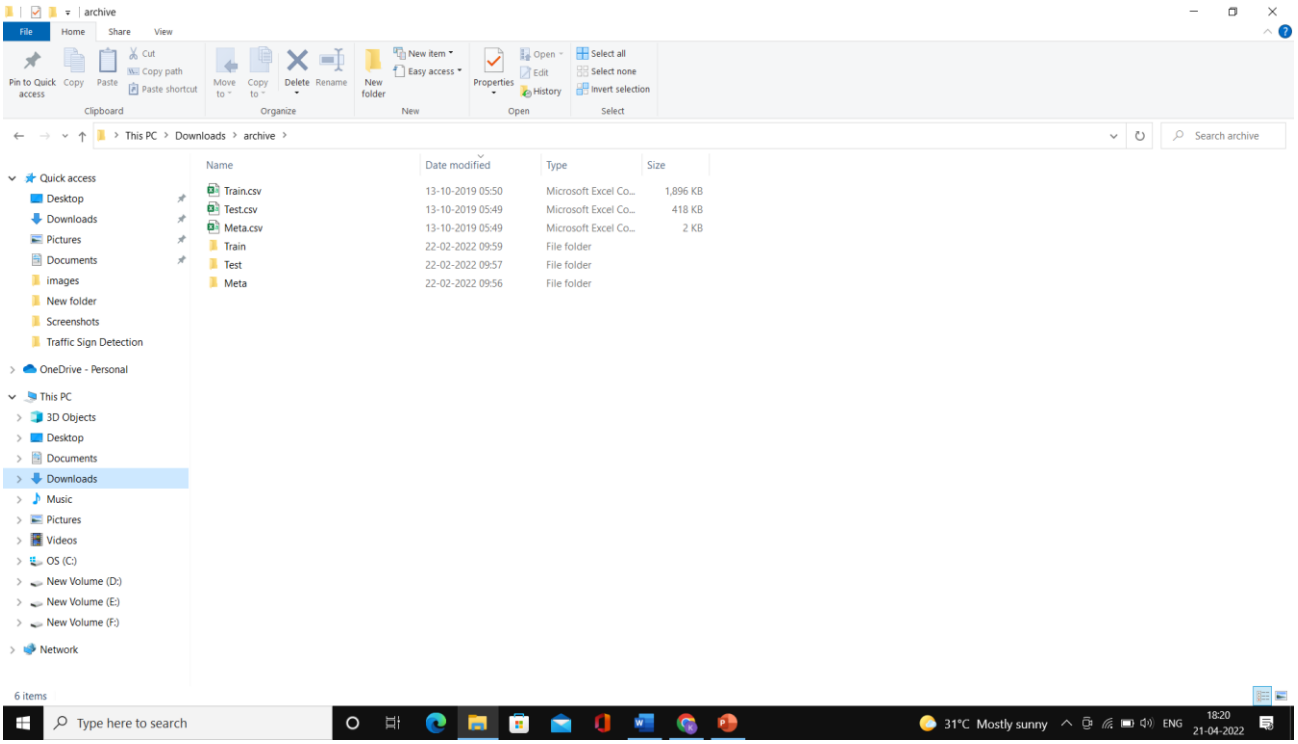
Model testing is referred to as the process where the performance of a fully trained model is evaluated on a testing set. We tested the model with various inputs and we got the accuracy of 94.96 percent.

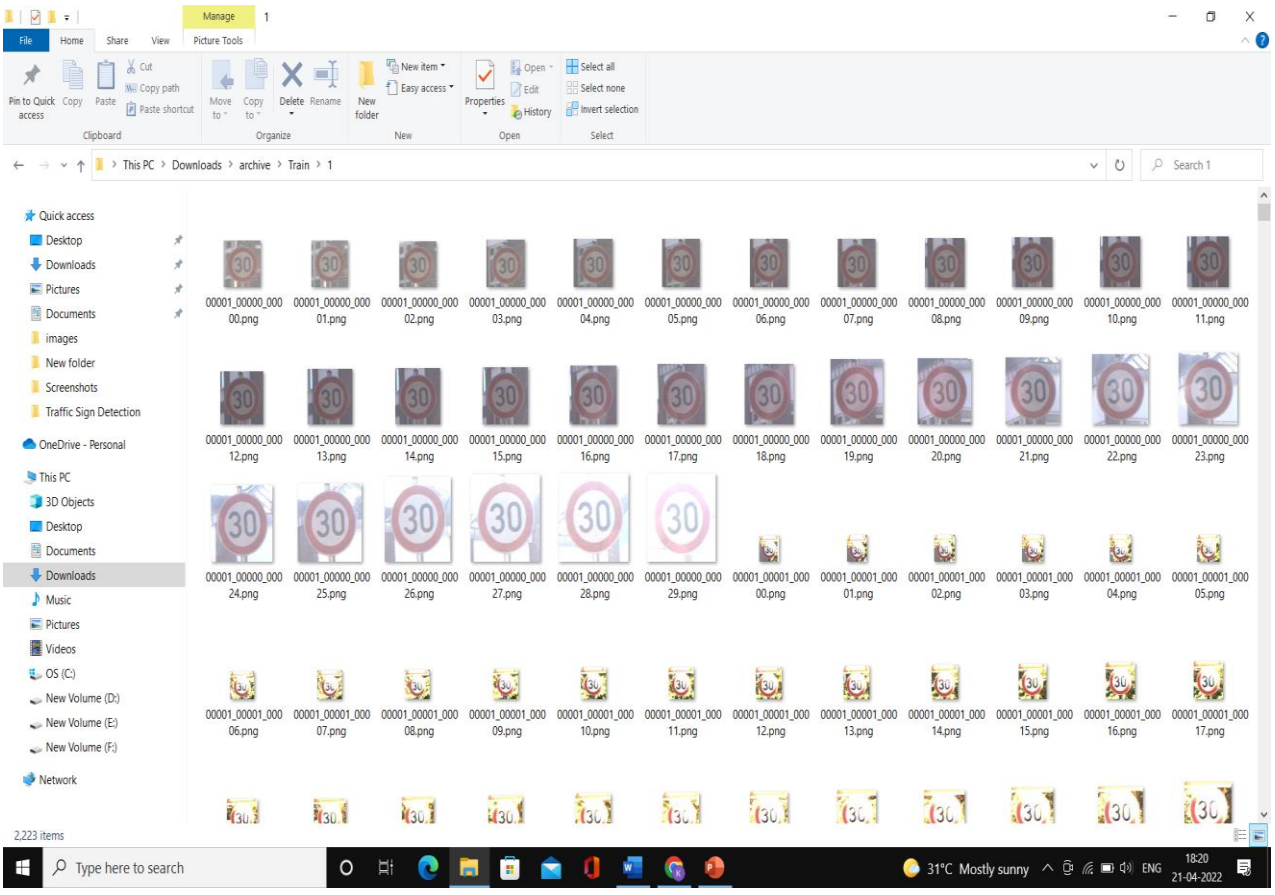
5. Model Deployment :-

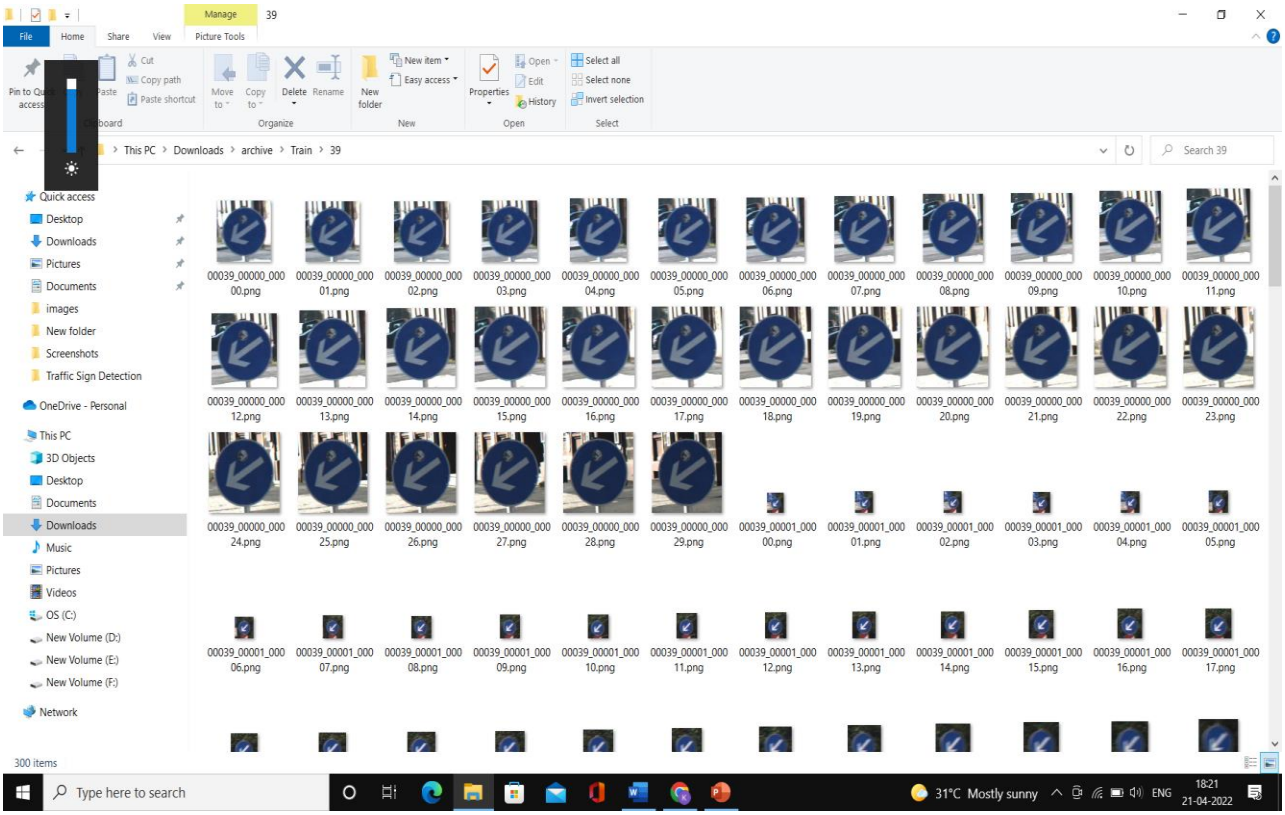
Deployment is the method by which you integrate a machine learning model into an existing production environment to make practical business decisions based on data. It is one of the last stages in the machine learning life cycle and can be one of the most cumbersome.

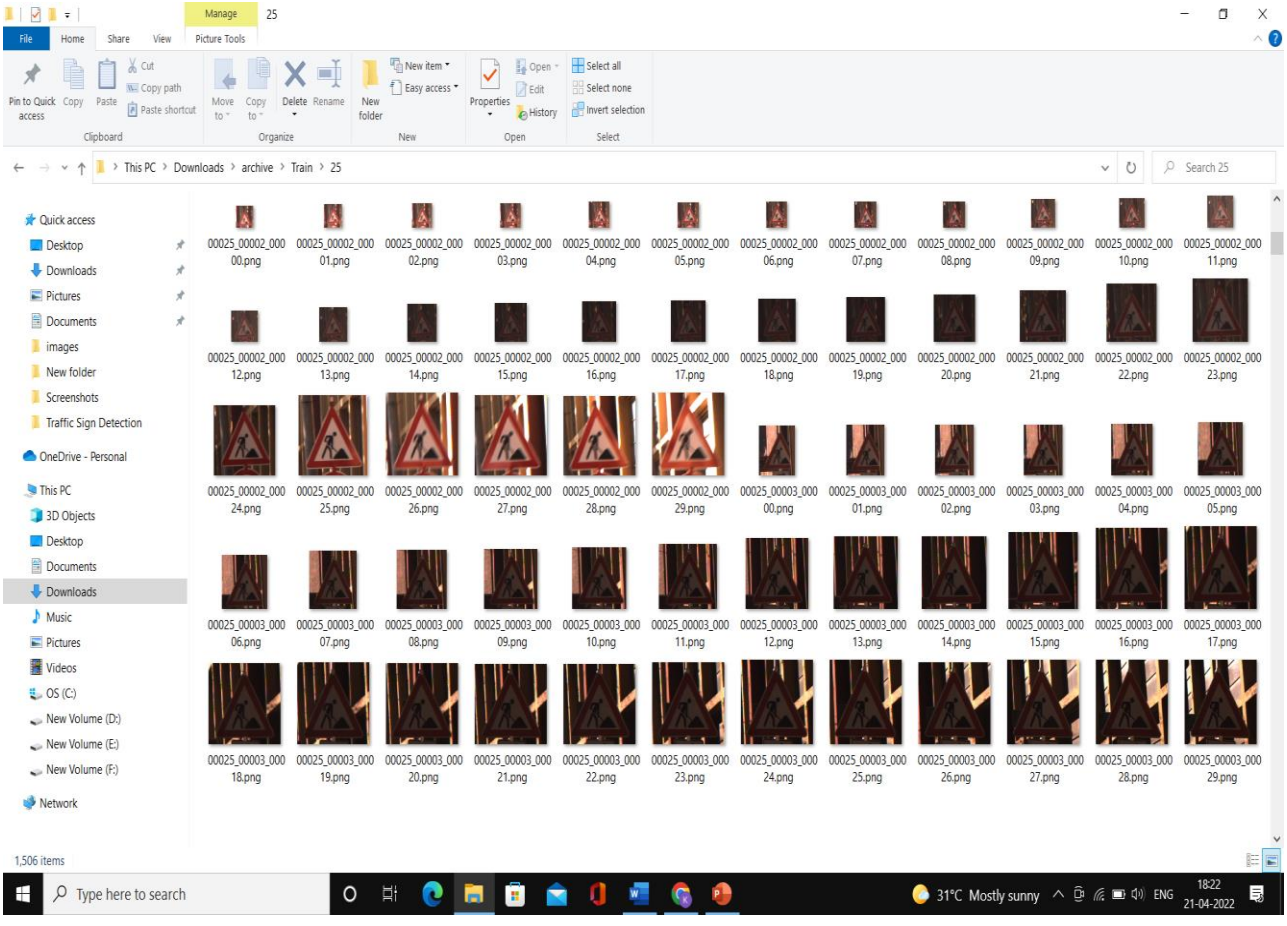
Deployment of the model is simply putting models into production, means making your models available to other systems. We used this model in our GUI python file for better experience of the user.

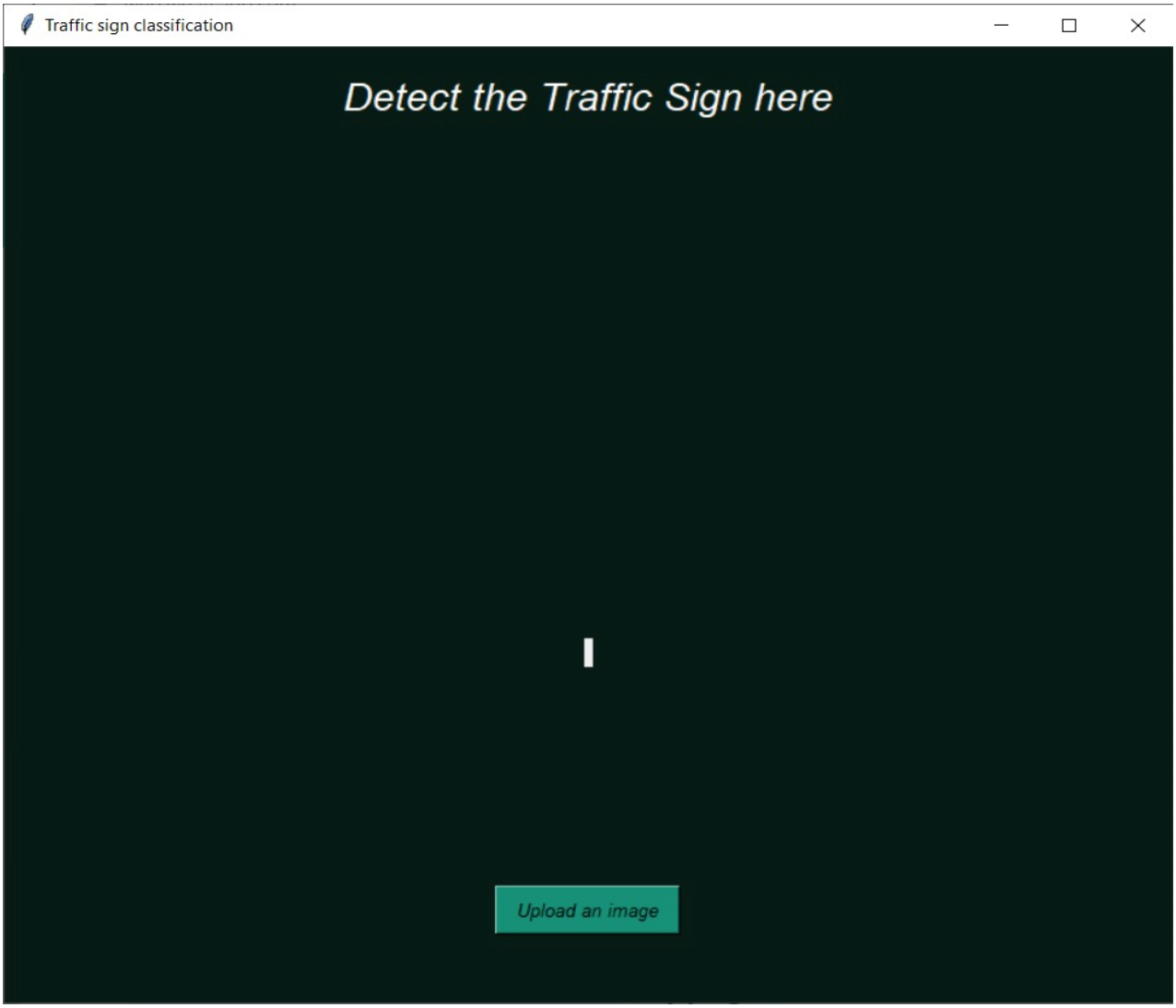
5.3 Screens

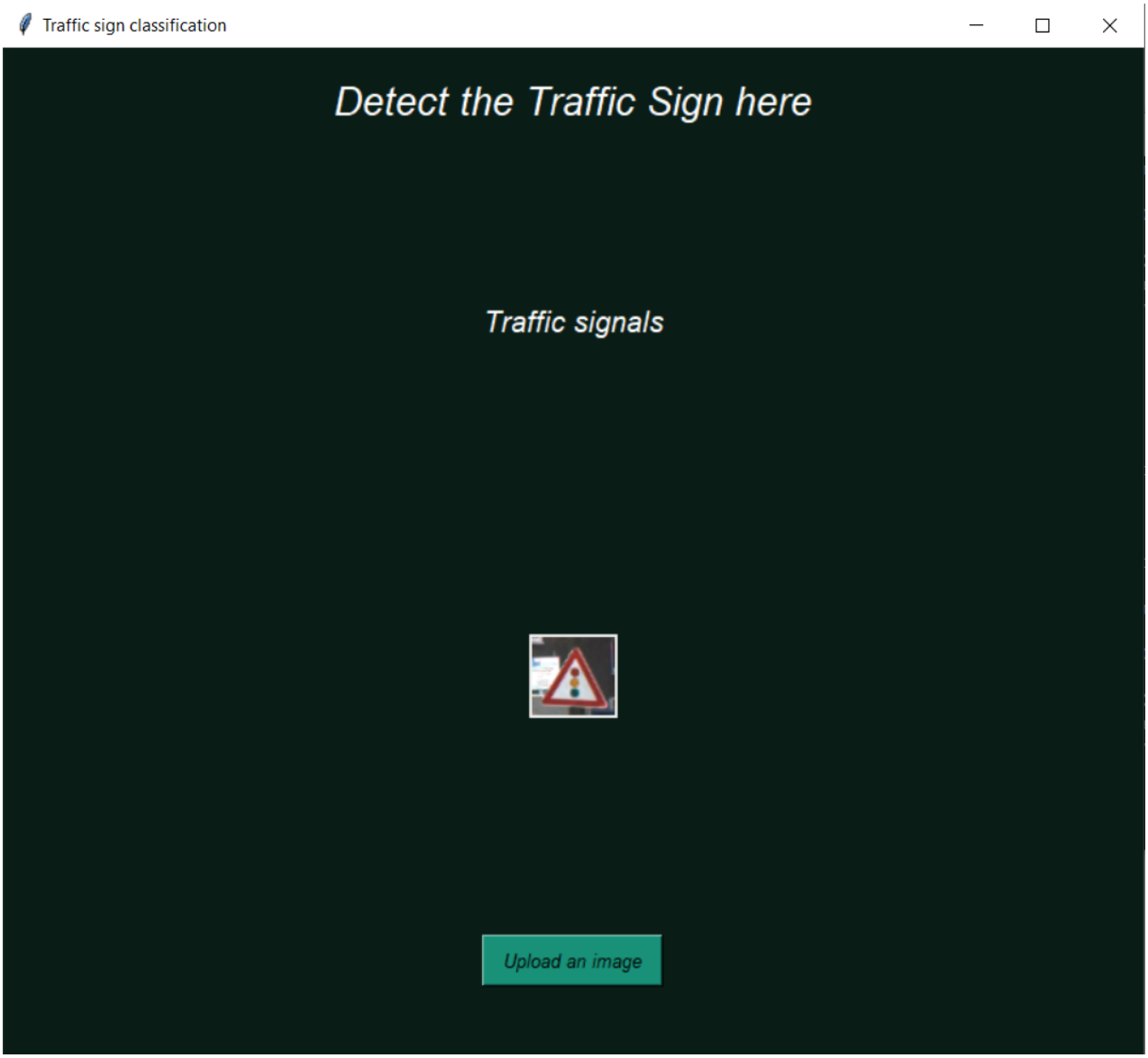

archive
This PC > Downloads > archive >
Search archive
Name
Date modified
Type
Size
Train.csv
Test.csv
Meta.csv
Train
Test
Meta
6 items
Type here to search
31°C Mostly sunny
ENG
18:20
21-04-2022

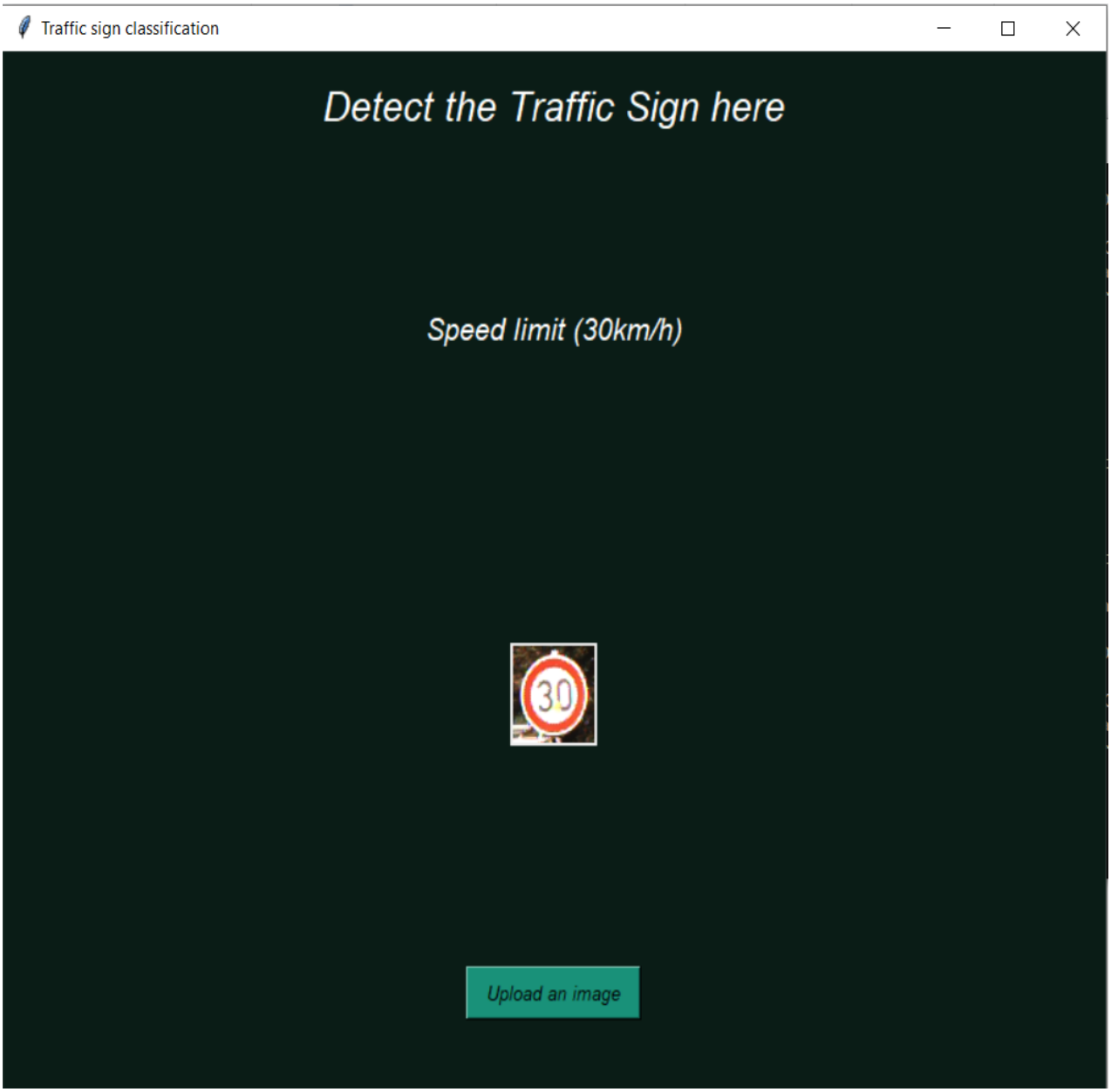

Manage
Picture Tools
This PC > Downloads > archive > Train > 1
Search 1
2,223 items
Type here to search
31°C Mostly sunny
ENG
18:20
21-04-2022


39
This PC > Downloads > archive > Train > 39
Search 39
300 items
Type here to search
31°C Mostly sunny
ENG
18:21
21-04-2022


25
This PC > Downloads > archive > Train > 25
Search 25
1,506 items
Type here to search
31°C Mostly sunny
ENG
18:22
21-04-2022


Traffic sign classification
Detect the Traffic Sign here
Upload an image


Traffic sign classification
Detect the Traffic Sign here
Traffic signals
Upload an image


Traffic sign classification
Detect the Traffic Sign here
Speed limit (30km/h)
Upload an image

6. SYSTEM TESTING

6.1 Introduction

System Testing is a type of software testing that is performed on a complete integrated system to evaluate the compliance of the system with the corresponding requirements. In system testing, integration testing passed components are taken as input. The goal of integration testing is to detect any irregularity between the units that are integrated together.

System testing detects defects within both the integrated units and the whole system. The result of system testing is the observed behaviour of a component or a system when it is tested.

System Testing is carried out on the whole system in the context of either system requirement specifications or functional requirement specifications or in the context of both.

System testing tests the design and behaviour of the system and also the expectations of the customer. It is performed to test the system beyond the bounds mentioned in the software requirements specification (SRS).

System Testing is basically performed by a testing team that is independent of the development team that helps to test the quality of the system impartial. It has both functional and non-functional testing.

6.2 Testing Methods

White Box Testing :-

White box testing is a software evaluating method used to examine the internal structure, design, coding and inner-working of software. Developers use this testing method to verify the flow of inputs and outputs through the application, improving usability and design and strengthening security. The concept is called "white box" because it is symbolically see-through, as the code is visible to the tester during the examination.

In comparison, when the inner code isn't visible, it is called black box testing. It is one of two parts of the Box Testing approach to software testing. White box testing in software engineering is based on the inner workings of an application and revolves around internal testing. The term “White Box” was used because of the see-through box concept. The clear box or White Box name symbolizes the ability to see through the software’s outer shell (or “box”) into its inner workings.

Generic steps of white box testing :-

- Design all test scenarios, test cases and prioritize them according to high priority number.
- This step involves the study of code at runtime to examine the resource utilization, not accessed areas of the code, time taken by various methods and operations and so on.
- In this step testing of internal subroutines takes place. Internal subroutines such as non-public methods, interfaces are able to handle all types of data appropriately or not.
- This step focuses on testing of control statements like loops and conditional statements to check the efficiency and accuracy for different data inputs.
- In the last step white box testing includes security testing to check all possible security loopholes by looking at how the code handles security.

Black Box Testing :-

Black box testing is a technique of software testing which examines the functionality of software without peering into its internal structure or coding. The primary source of black box testing is a specification of requirements that is stated by the customer.

In this method, tester selects a function and gives input value to examine its functionality, and checks whether the function is giving expected output or not. If the function

produces correct output, then it is passed in testing, otherwise failed. The test team reports the result to the development team and then tests the next function. After completing testing of all functions if there are severe problems, then it is given back to the development team for correction.

Generic steps of black box testing :-

- The black box test is based on the specification of requirements, so it is examined in the beginning.
- In the second step, the tester creates a positive test scenario and an adverse test scenario by selecting valid and invalid input values to check that the software is processing them correctly or incorrectly.
- In the third step, the tester develops various test cases such as decision table, all pairs test, equivalent division, error estimation, cause-effect graph, etc.
- The fourth phase includes the execution of all test cases.
- In the fifth step, the tester compares the expected output against the actual output.
- In the sixth and final step, if there is any flaw in the software, then it is cured and tested again.

Unit Testing :-

Unit testing involves the testing of each unit or an individual component of the software application. It is the first level of functional testing. The aim behind unit testing is to validate unit components with its performance. A unit is a single testable part of a software system and tested during the development phase of the application software.

The purpose of unit testing is to test the correctness of isolated code. A unit component is an individual function or code of the application. White box testing approach used for unit testing and usually done by the developers.

Whenever the application is ready and given to the Test engineer, he/she will start checking every component of the module or module of the application independently or one by one, and this process is known as Unit testing or components testing.

In a testing level hierarchy, unit testing is the first level of testing done before integration and other remaining levels of the testing. It uses modules for the testing process which reduces the dependency of waiting for Unit testing frameworks, stubs, drivers and mock objects are used for assistance in unit testing.

Integration Testing :-

Integration testing is the second level of the software testing process comes after unit testing. In this testing, units or individual components of the software are tested in a group. The focus of the integration testing level is to expose defects at the time of interaction between integrated components or units.

Unit testing uses modules for testing purpose, and these modules are combined and tested in integration testing. The Software is developed with a number of software modules that are coded by different coders or programmers. The goal of integration testing is to check the correctness of communication among all the modules.

Once all the components or modules are working independently, then we need to check the data flow between the dependent modules is known as integration testing.

Validation Testing :-

Validation testing is testing where tester performed functional and non-functional testing. Here functional testing includes Unit Testing (UT), Integration Testing (IT) and System Testing (ST), and non-functional testing includes User acceptance testing (UAT).

Validation testing is also known as dynamic testing, where we are ensuring that "we have developed the product right." And it also checks that the software meets the business needs of the client.

6.3 Test Cases :-

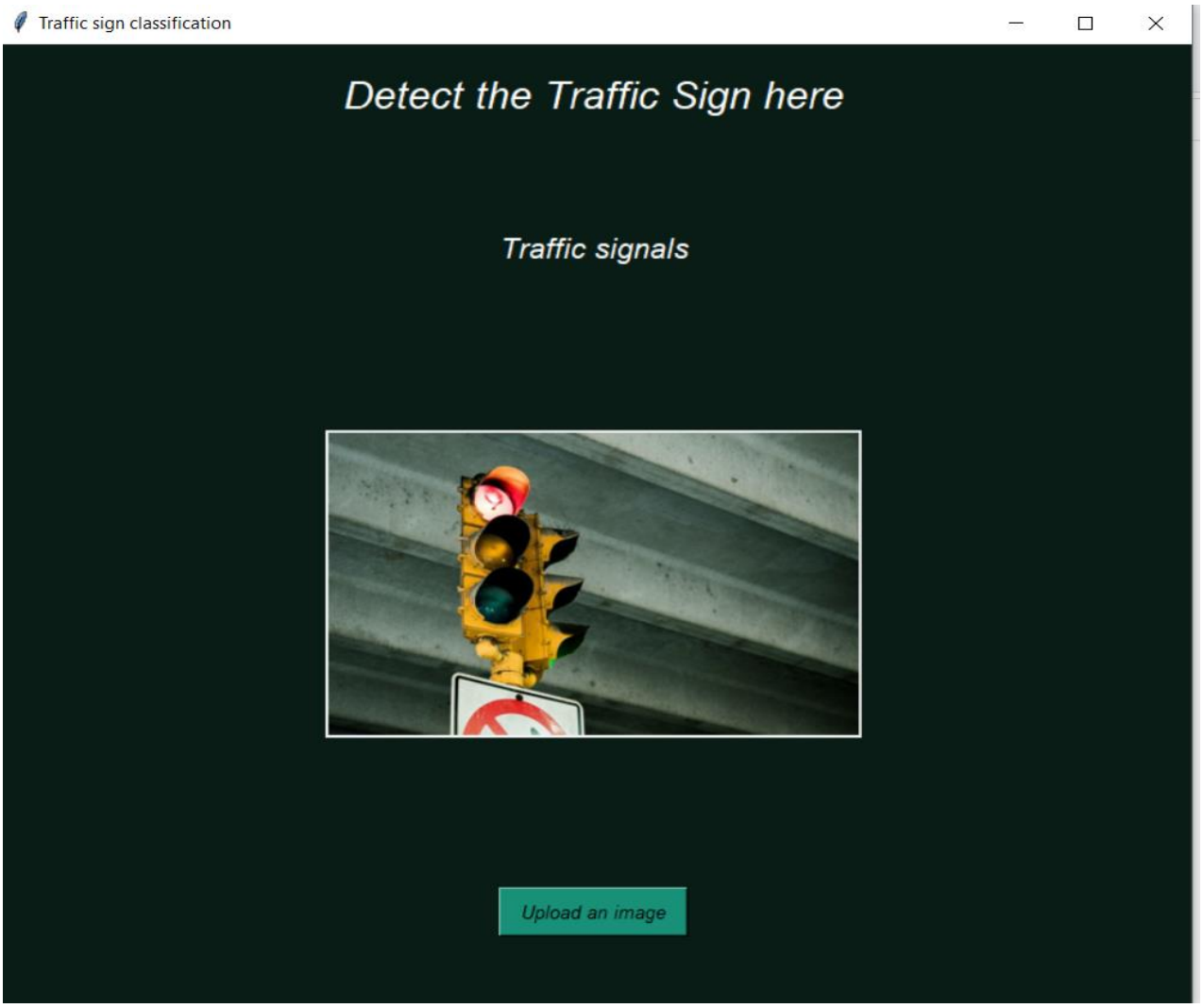


Figure 1

When we upload an image which include traffic signals as traffic sign, it displays text as traffic signals and also gives audio as traffic signals.

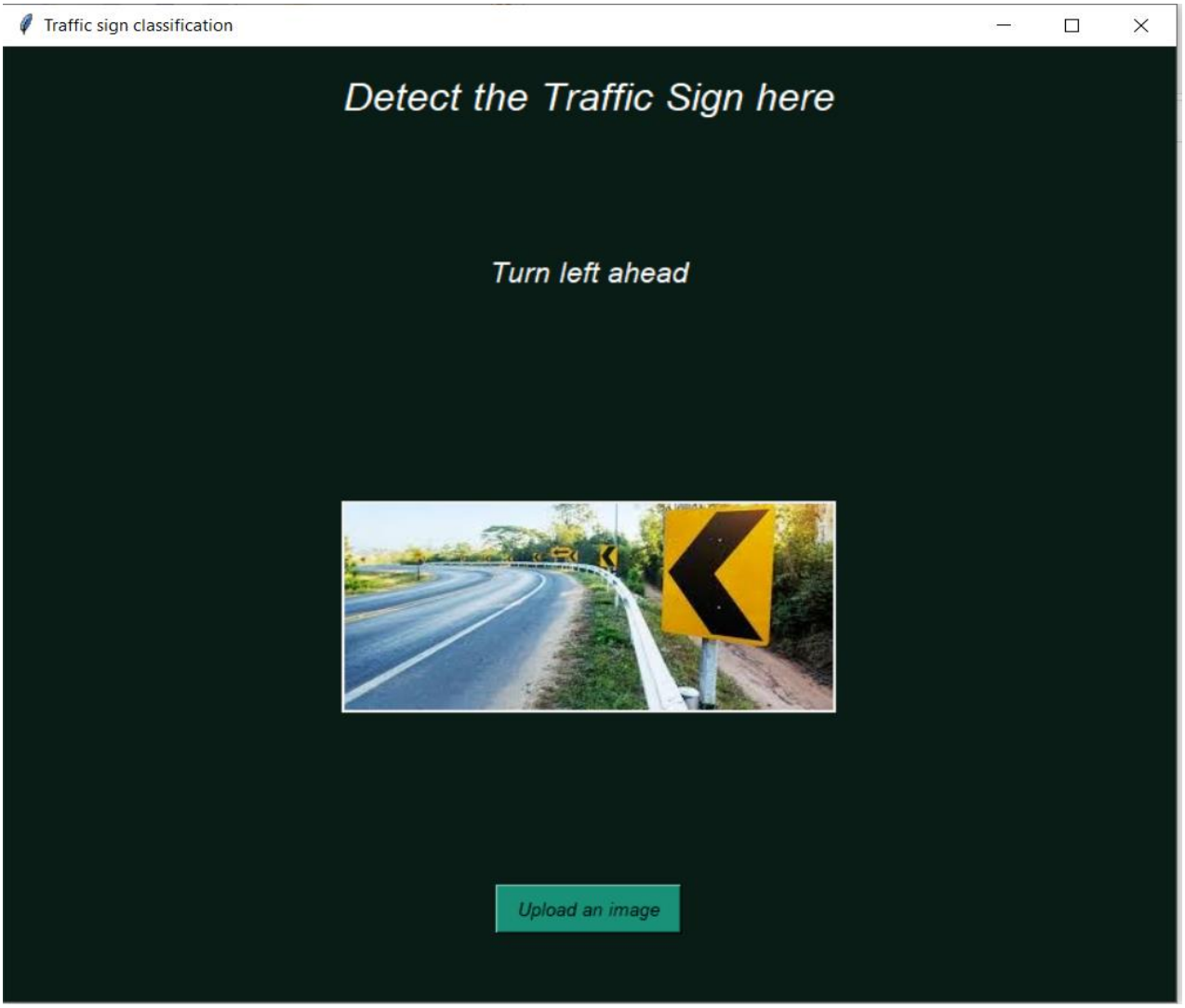


Figure 2

When we upload an image which include turn left ahead as traffic sign, it displays text as Turn left ahead and also gives audio as Turn left ahead.

7. CONCLUSION AND FUTURE SCOPE

Conclusion :-

Through this work, a model for traffic sign recognition system is successfully implemented using convolutional neural networks, needed for vehicles as a measure for ensuring road safety. The whole task is implemented on google colab and spyder notebook using Python programming for machine learning and its strong libraries of deep learning. The parameters of the designed neural network model are finely tuned in order to get good accuracy results with an accuracy of 94.96%.

Important point to be noted is that taking a large number of CNNs can increase learning rate of model and images for training are generally augmented but real time image capturing can't be augmented much fast so quick reliability is needed. Extraction of image's feature can be improved along with understanding sign recognition and related features like driving behaviour for a responsive system for high and low level of traffic.

Future Scope :-

Another direction for further research is to develop a real time traffic sign recognition system which captures a video by a camera mounted on the vehicle, detects and recognises the traffic signs in real time and gives the result to the driver within a sufficient time frame in order to take the right action.

The crucial issue in real time applications is the time spent to recognise the traffic sign. This should be reduced to the 169 minimum by choosing the proper techniques for real time applications and by optimising the code. The methods presented in this thesis can be modified to fit the real time requirements.

8. BIBLIOGRAPHY

- https://www.researchgate.net/publication/224255280_Real-time_traffic_sign_recognition_system
- https://www.ijert.org/traffic-sign-recognition-using-machine-learning-a-review?amp=1

9. APPENDIX

9.1 Introduction to Python

Python is an easy to learn, powerful programming language. It has efficient high-level data structures and a simple but effective approach to object-oriented programming. Python's elegant syntax and dynamic typing, together with its interpreted nature, make it an ideal language for scripting and rapid application development in many areas on most platforms.

The Python interpreter is easily extended with new functions and data types implemented in C or C++ (or other languages callable from C). Python is also suitable as an extension language for customizable applications.

Features in Python :-

Easy to code :-

Python is a high-level programming language. Python is very easy to learn the language as compared to other languages like C, C#, Javascript, Java, etc. It is very easy to code in python language and anybody can learn python basics in a few hours or days. It is also a developer-friendly language.

Free and Open Source :-

Python language is freely available at the official website and you can download it from the given download link below click on the Download Python keyword. Since it is open-source, this means that source code is also available to the public. So, you can download it as, use it as well as share it.

Object-Oriented Language :-

One of the key features of python is Object-Oriented programming. Python supports object-oriented language and concepts of classes, objects encapsulation, etc.

GUI Programming Support :-

Graphical User interfaces can be made using a module such as PyQt5, PyQt4, wxPython, or Tk in python.
PyQt5 is the most popular option for creating graphical apps with Python.

High-level Language :-

Python is a high-level language. When we write programs in python, we do not need to remember the system architecture, nor do we need to manage the memory.

Extensible feature :-

Python is an Extensible language. We can write us some Python code into C or C++ language and also we can compile that code in C/C++ language.

Python is Portable language :-

Python language is also a portable language. For example, if we have python code for windows and if we want to run this code on other platforms such as Linux, Unix, and Mac then we do not need to change it, we can run this code on any platform.

Python is Integrated language :-

Python is also an Integrated language because we can easily integrated python with other languages like c, c++, etc.

Interpreted Language :-

Python is an Interpreted Language because Python code is executed line by line at a time. like other languages C, C++, Java, etc. there is no need to compile python code this makes it easier to debug our code. The source code of python is converted into an immediate form called bytecode.

Large Standard Library :-

Python has a large standard library which provides a rich set of module and functions so you do not have to write your own code for every single thing. There are many libraries present in python for such as regular expressions, unit-testing, web browsers, etc.

Dynamically Typed Language :-

Python is a dynamically-typed language. That means the type (for example- int, double, long, etc.) for a variable is decided at run time not in advance because of this feature we don’t need to specify the type of variable.

Libraries Used :-

Numpy :-

NumPy, which stands for Numerical Python, is a library consisting of multidimensional array objects and a collection of routines for processing those arrays. Using NumPy, mathematical and logical operations on arrays can be performed.

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

Pandas :-

Pandas is an open source library in Python. It provides ready to use high-performance data structures and data analysis tools. Pandas module runs on top of NumPy and it is popularly used for data science and data analytics.

NumPy is a low-level data structure that supports multi-dimensional arrays and a wide range of mathematical array operations. Pandas has a higher-level interface. It also provides streamlined alignment of tabular data and powerful time series functionality.

Data Frame is the key data structure in Pandas. It allows us to store and manipulate tabular data as a 2-D data structure.

Matplotlib :-

Matplotlib is a cross-platform, data visualization and graphical plotting library for Python and its numerical extension NumPy. As such, it offers a viable open source alternative to MATLAB. Developers can also use matplotlib's APIs (Application Programming Interfaces) to embed plots in GUI applications.

TensorFlow :-

TensorFlow is a free and open-source library for creating machine learning models. It is a fantastic platform for everyone interested in working with machine learning and artificial intelligence.

TensorFlow is an end-to-end open-source machine learning platform with a focus on deep neural networks. Deep learning is a subtype of machine learning that analyses massive amounts of unstructured data. Since it works with structured data, deep learning is different from normal machine learning.

TensorFlow provides a diverse and complete set of libraries, tools, and community resources. It allows developers to create and deploy state-of-the-art machine learning-powered applications. One of the most appealing aspects of TensorFlow is that it makes use of Python to create a nice front-end API for developing applications that run in high-performance, optimized C++.

PIL :-

PIL is the Python Imaging Library which provides the python interpreter with image editing capabilities. The Image module provides a class with the same name which is used to represent a PIL image. The module also provides a number of factory functions, including functions to load images from files, and to create new images.

OS :-

The OS module in Python provides functions for interacting with the operating system. OS comes under Python's standard utility modules. This module provides a portable way of using operating system-dependent functionality. The os and os.path modules include many functions to interact with the file system.

Sklearn :-

Scikit-learn (Sklearn) is the most useful and robust library for machine learning in Python. It provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction via a consistence interface in Python. This library, which is largely written in Python, is built upon NumPy, SciPy and Matplotlib.

Keras :-

Keras is a minimalist Python library for deep learning that can run on top of Theano or TensorFlow. It was developed to make implementing deep learning models as fast and easy as possible for research and development.

It runs on Python 2.7 or 3.5 and can seamlessly execute on GPUs and CPUs given the underlying frameworks. It is released under the permissive MIT license.

Tkinter :-

Tkinter is the de facto way in Python to create Graphical User interfaces (GUIs) and is included in all standard Python Distributions. In fact, it's the only framework built into the Python standard library.

This Python framework provides an interface to the Tk toolkit and works as a thin object-oriented layer on top of Tk. The Tk toolkit is a cross-platform collection of 'graphical control elements', aka widgets, for building application interfaces.

pyttsx3 :-

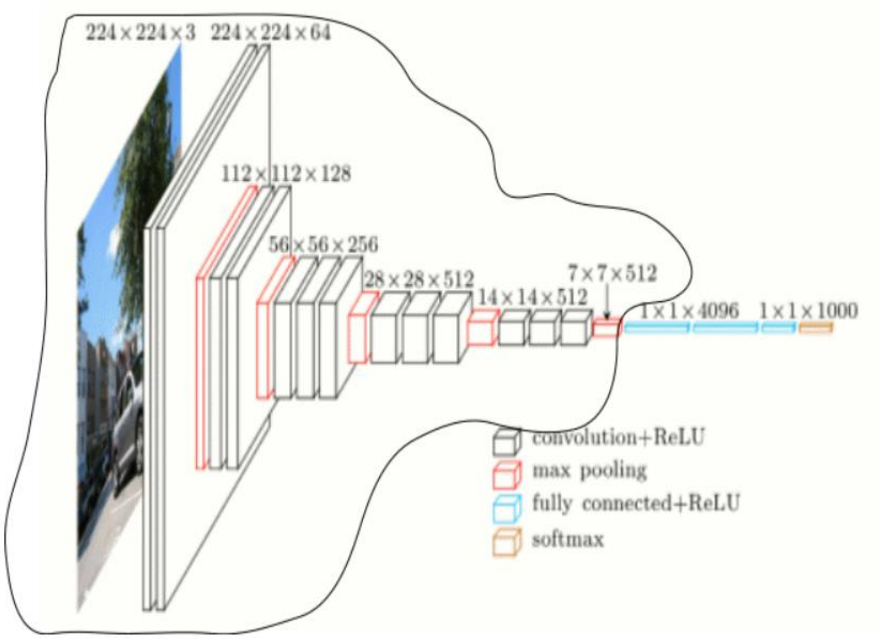
pyttsx3 is a text-to-speech conversion library in Python. Unlike alternative libraries, it works offline and is compatible with both Python 2 and 3. An application invokes the pyttsx3.init() factory function to get a reference to a pyttsx3.

9.2 Introduction to CNN

Convolutional neural networks refer to a sub-category of neural networks: they, therefore, have all the characteristics of neural networks. However, CNN is specifically designed to process input images. Their architecture is then more specific: it is composed of two main blocks.

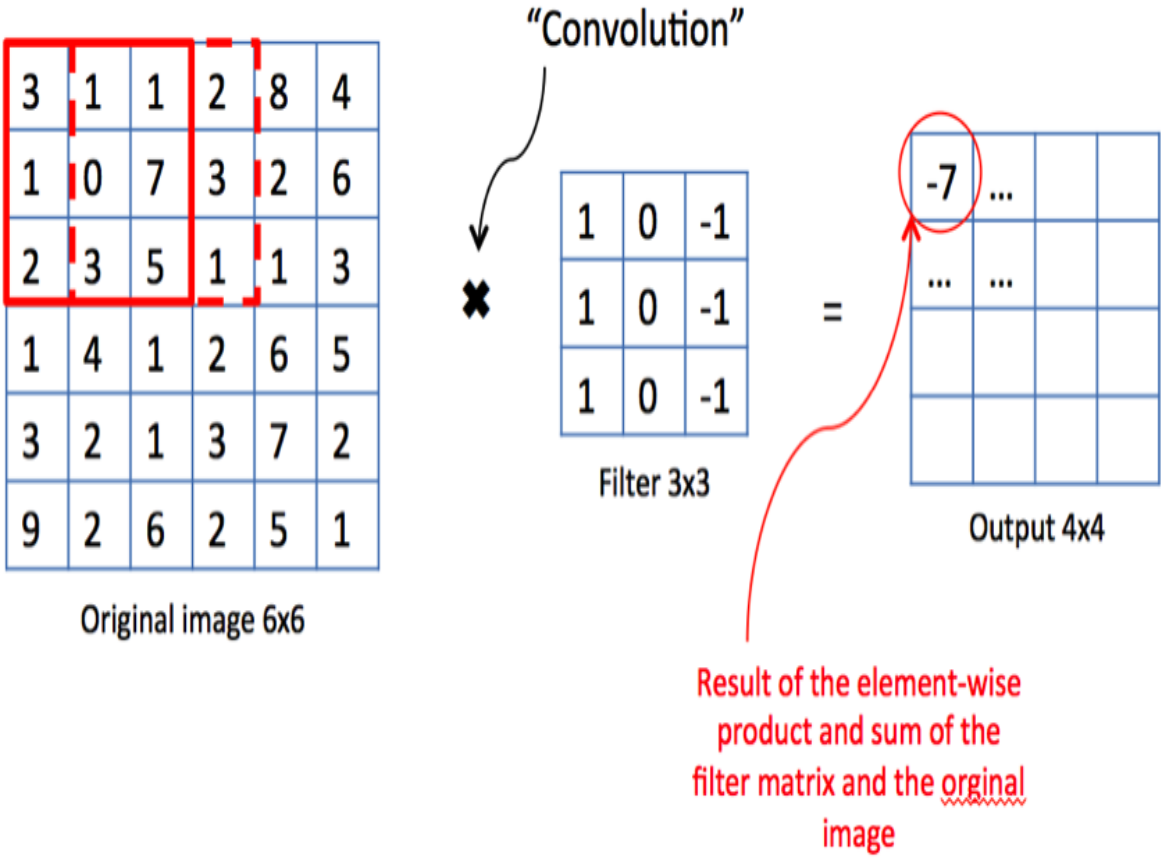
The first block makes the particularity of this type of neural network since it functions as a feature extractor. To do this, it performs template matching by applying convolution filtering operations. The first layer filters the image with several convolution kernels and returns "feature maps", which are then normalized (with an activation function) and/or resized.

This process can be repeated several times: we filter the features maps obtained with new kernels, which gives us new features maps to normalize and resize, and we can filter again, and so on. Finally, the values of the last feature maps are concatenated into a vector. This vector defines the output of the first block and the input of the second.



The second block is not characteristic of a CNN: it is in fact at the end of all the neural networks used for classification. The input vector values are transformed (with several linear combinations and activation functions) to return a new vector to the output. This last vector contains as many elements as there are classes: element i represents the probability that the image belongs to class i.

Each element is therefore between 0 and 1, and the sum of all is worth 1. These probabilities are calculated by the last layer of this block (and therefore of the network), which uses a logistic function (binary classification) or a softmax function (multi-class classification) as an activation function.



As with ordinary neural networks, the parameters of the layers are determined by gradient backpropagation: the cross-entropy is minimized during the training phase. But in the case of CNN, these parameters refer in particular to the image features.

Advantages:

- Very High accuracy in image recognition problems.
- Automatically detects the important features without any human supervision.
- Weight sharing.

Different layers of a CNN

There are four types of layers for a convolutional neural network: the convolutional layer, the pooling layer, the ReLU correction layer and the fully-connected layer.

The convolutional layer :-

The convolutional layer is the key component of convolutional neural networks, and is always at least their first layer.

Its purpose is to detect the presence of a set of features in the images received as input. This is done by convolution filtering: the principle is to "drag" a window representing the feature on the image, and to calculate the convolution product between the feature and each portion of the scanned image. A feature is then seen as a filter: the two terms are equivalent in this context.

The convolutional layer thus receives several images as input, and calculates the convolution of each of them with each filter. The filters correspond exactly to the features we want to find in the images.

The Pooling layer :-

This type of layer is often placed between two layers of convolution: it receives several feature maps and applies the pooling operation to each of them. The pooling operation consists in reducing the size of the images while preserving their important characteristics**.**

To do this, we cut the image into regular cells, then we keep the maximum value within each cell. In practice, small square cells are often used to avoid losing too much information. The most common choices are 2x2 adjacent cells that don't overlap, or 3x3 cells, separated from each other by a step of 2 pixels (thus overlapping).

We get in output the same number of feature maps as input, but these are much smaller. The pooling layer reduces the number of parameters and calculations in the network**.** This improves the efficiency of the network and avoids over-learning.

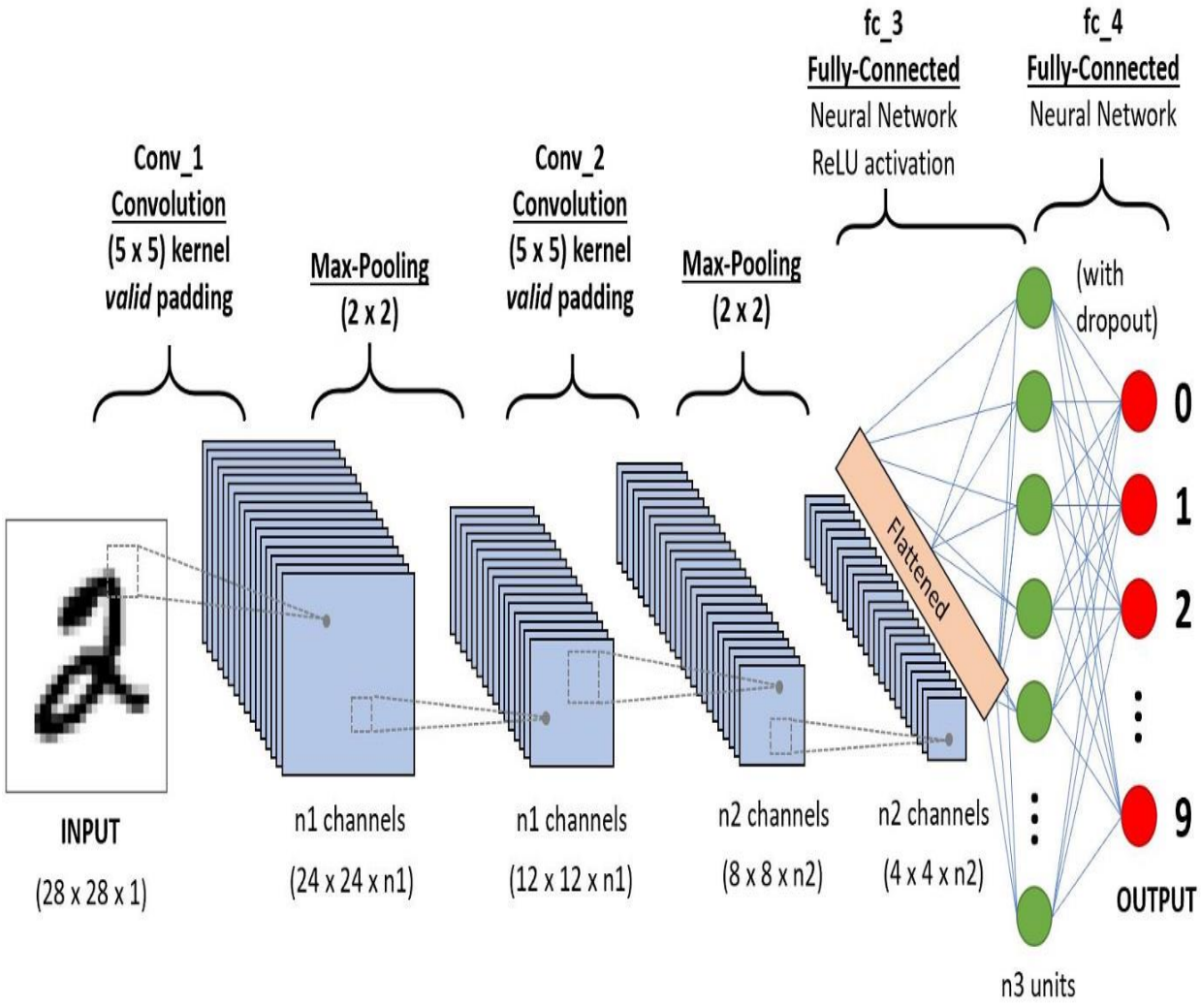
The maximum values are spotted less accurately in the feature maps obtained after pooling than in those received in input — this is a big advantage! For example, when you want to recognize a dog, its ears do not need to be located as precisely as possible: knowing that they are located almost next to the head is enough!

The ReLU Correction layer :-

In this layer we remove every negative value from the filtered image and replace it with zero. This function only activates when the node input is above a certain quantity. So, when the input is below zero the output is zero.

However, when the input rises above a certain threshold it has *linear relationship* with the dependent variable. This means that it is able to *accelerate the speed of* a training data set in a deep neural network that is faster than other activation functions – this is done to avoid summing up with zero.

ReLU (Rectified Linear Units) refers to the real non-linear function defined by *ReLU(x)=max(0,x)*. The ReLU correction layer replaces all negative values received as inputs by zeros. It acts as an activation function**.**



The fully-connected layer :-

The fully-connected layer is always the last layer of a neural network, convolutional or not — so it is not characteristic of a CNN.

This type of layer receives an input vector and produces a new output vector. To do this, it applies a linear combination and then possibly an activation function to the input values received.

The last fully-connected layer classifies the image as an input to the network: it returns a vector of size N, where N is the number of classes in our image classification problem. Each element of the vector indicates the probability for the input image to belong to a class.

To calculate the probabilities, the fully-connected layer, therefore, multiplies each input element by weight, makes the sum, and then applies an activation function (logistic if N=2, softmax if N>2).

This is equivalent to multiplying the input vector by the matrix containing the weights. The fact that each input value is connected with all output values explains the term fully connected.