A

Major Project Report on

A MULTI PURPOSE ROAD SURVEILLANCE SYSTEM WITH DEEP LEARNING

Submitted for partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

in

ELECTRONICS AND COMMUNICATION ENGINEERING

Submitted by

M.BHAVANA	**20K85A0426**
G.VIKRAM REDDY	**20K85A0428**
N.ROOP PAVAN	**20K85A0432**
K.P.S.KAUSHIK	**20K85A0435**

Under the Guidance of

Mrs. P. KANTHA RATNAM

Assistant Professor

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

St. MARTIN'S ENGINEERING COLLEGE

UGC Autonomous

Affiliated to JNTUH, Approved by AICTE,

Accredited by NBA & NAAC A+, ISO 9001: 2008 Certified

Dhulapally, Secunderabad – 500100.

MAY-2023

St. MARTIN'S ENGINEERING COLLEGE

UGC Autonomous

NBA & NAAC A+ Accredited

Dhulapally, Secunderabad-500100

CERTIFICATE

This is to certify that the major project entitled '**A MULTI PURPOSE ROAD SURVEILLANCE SYSTEM WITH DEEP LEARNING'** is being submitted by M.BHAVANA 20K85A0426, G.VIKRAM REDDY 20K85A0428, N.ROOP PAVAN 20K85A0432, K.P.S.KAUSHIK 20K85A0435 in partial fulfillment of the requirement for the award of degree of **BACHELOR OF TECHNOLOGY** in **Electronics and Communication Engineering.** It is a record of bonafide work carried out by them. The result embodied in this report has been verified and found satisfactory.

Project Guide	**Head of the Department**
Mrs. P. Kantha Ratnam	**Dr. B. Hari Krishna**
Assistant Professor	Professor & HOD

Internal Examiner	**External Examiner**

St. MARTIN'S ENGINEERING COLLEGE

UGC Autonomous

NBA & NAAC A+ Accredited

Dhulapally, Secunderabad-500100

DECLARATION

We, the students of **Bachelor of Technology** in Department of **Electronics and Communication Engineering,** session: **2019-2023, St. Martin's Engineering College, Dhulapally, Kompally, Secunderabad** hereby declare that the work presented in this project entitled '**A MULTI PURPOSE ROAD SURVEILLANCE SYSTEM WITH DEEP LEARNING '** is the outcome of our own bonafide work and it is correct to the best of our knowledge and this work has been undertaken taking care of Engineering Ethics. The result embodied in this project report has not been submitted in any university for award of any degree.

M.BHAVANA	**20K85A0426**
G.VIKRAM REDDY	**20K85A0428**
N.ROOP PAVAN	**20K85A0432**
K.P.S.KAUSHIK	**20K85A0435**

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, Secunderabad for permitting us to undertake this project.

We are also thankful to Professor **Dr. B. HARI KRISHNA**, Head of the Department, Electronics and Communication Engineering, St. Martin's Engineering College, Dhulapally, Secunderabad for his support and guidance throughout our project as well as Project Coordinator **Mr. G. RAMESH**, Assistant Professor, Electronics and Communication Engineering for his valuable support.

We would like to express our sincere gratitude and indebtedness to our project guide **Mrs. P. KANTHA RATNAM**, Assistant Professor, Department of Electronics and Communication Engineering, St. Martins Engineering College, Dhulapally for his support and guidance throughout our project.

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

M.BHAVANA	**20K85A0426**
G.VIKRAM REDDY	**20K85A0428**
N.ROOP PAVAN	**20K85A0432**
K.P.S.KAUSHIK	**20K85A0435**

ABSTRACT

The traffic surveillance system is accumulated with an enormous amount of data regarding road traffic each and every second. Monitoring these data with the human eye is a tedious process and it also requires manpower for monitoring. Deep learning approach (Convolutional Neural Network) can be utilized for traffic monitoring and control. The traffic surveillance data are pre-processed to construct the training dataset. The Traffic net is constructed by transferring the network to traffic applications and retraining it with self-established data set. This Traffic net can be used for regional detection in large scale applications. Further, it can be implemented across-the-board. The efficiency is admirably verified through speedy discovery in the high accuracy in the case study. The tentative assessment could pull out to its successful application to a traffic surveillance system and has potential enrichment for the intelligent transport system in future.

The existing method requires a huge amount of hardware equipment's deployed to the road. Moreover, they are very sensitive to the external noise and environmental conditions. It is more accurate when processing a limited number of vehicles, but it does not work well on large scale dataset.

The purpose of the proposed work is to make a speedy traffic detection system which reduces the manpower and detected Multiclass problems namely fire detection, accident detection, dense and sparse traffic detection. The main aim is to classify the given input image to dense or sparse based on the trained model from the input dataset.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ACRONYMS AND DEFINITIONS

S.No	Acronym	Definition
01	SVM	Support Vector Machine
02	KNN	K-Nearest Neighbor
03	PNN	Probabilistic neural networks
04	RBF	Radial basis function
05	IDLE	Integerated Development and Learning Environment
06	GUI	Graphical user interface
07	UML	Unified Modelling Language
08	OMT	Object-modelling technique
09	OOSE	Object-oriented software engineering
10	GRNN	General Regression Neural Networks
11	RBF	Radial basis function
12	AI	Artificial Intelligence
13	ML	Machine Learning
14	DL	Deep Learning

CHAPTER 1

INTRODUCTION

1.1 Overview

As urbanization has accelerated, traffic in urban areas has increased significantly, and the similar phenomenon has been appeared in freeways connected to the urban areas as well. The real-time monitoring of traffic on freeways could provide sophisticated traffic information to drivers, so the drivers could choose alternative routes to avoid heavy traffic. Furthermore, long-term records of traffic monitoring will be helpful for developing efficient transportation policies and strategies across urban and suburban areas. Currently, the typical means of monitoring traffic information use closed circuit television (CCTV) or detection equipment. The detection equipment includes loop detectors, image detectors, dedicated short range communication (DSRC), and radar detectors. In general, CCTVs are installed at fixed locations, and they can monitor the area on the freeway 24 hours a day. CCTV can monitor only limited areas; therefore, multiple CCTV circuits are necessary to monitor a wide range of freeways. However, the installation and maintenance of the multiple CCTV circuits is costly. In addition, it is difficult to detect vehicles in CCTV videos automatically due to the overlapping between vehicles because CCTV usually captures freeways in an oblique direction.

Recently, to overcome the limitations of collecting traffic information through CCTV, video collection methods employing unmanned aerial vehicles (UAVs) are being used [6]. Unlike CCTV, a UAV can monitor a wide range of freeways by elevating its altitude or moving its location, and it can travel to a specific location to observe unexpected situations, such as traffic accidents. Furthermore, a UAV views the freeways in a perpendicular direction, so the vehicles in the recorded videos do not overlap. Currently, however, videos from installed CCTV or operated UAVs are monitored by humans. Therefore, as the number of CCTV circuits and UAVs increases, more human resources are required. Moreover, we can not avoid human error; it is highly demanding to analyze real-time videos to effectively monitor traffic information.

In contrast to CCTV videos taken at a fixed height, the altitude of UAV varies at every time the video is recorded, and sometimes the altitude of UAV changes during recording. If the image scales are not fixed, we are not able to estimate the vehicle’s traveling distance on the actual road by simply measuring moving distance of the vehicle in sequential images. Therefore, to determine the exact speed of a vehicle by tracking the vehicle in sequential images, the image scale of each image should be estimated and the changes in the image scale should be taken into account. For example, the scale of the image was obtained by comparing a pre-defined structure on an actual road with its corresponding object in the first frame of a video. This approach requires a pre-definition of a structure for each location; therefore, images without known structures cannot be utilized. Later, the image scale is calculated by comparing the average sizes of vehicles in the images and pre-measured and averaged actual vehicle size. Although these methods have somewhat resolved the restrictions associated with a UAV's flight area, the calculated image scale is not accurate because the size of vehicle varies depend on the types of the vehicles. For instance, a detected vehicle can include sedans, vans, buses, of trucks.

Motivation

Deep learning approach (Convolutional Neural Network) can be utilized for traffic monitoring and control. The traffic surveillance data are pre-processed to construct the training dataset. The Traffic net is constructed by transferring the network to traffic applications and retraining it with self-established data set.

1.1.1 Problem Definition

The Existing method requires a huge amount of Hardware equipments deployed to the Road. Moreover, they are very sensitive to the external noise and environmental conditions. It is more accurate when processing a limited number of vehicles but it does not work well on large scale dataset.

1.1.2 Objective of Product

The purpose of the proposed work is to make a speedy traffic detection system which reduces the man power and detected Multiclass problems namely fire detection, accident detection, dense and sparse traffic detection. The main aim is to classify the given input image to dense or sparse based on the trained model from the input dataset.

Deep learning is the subset of AI (Artificial Intelligence) and ML (MachineLearning) as shown in below figure 1.1.

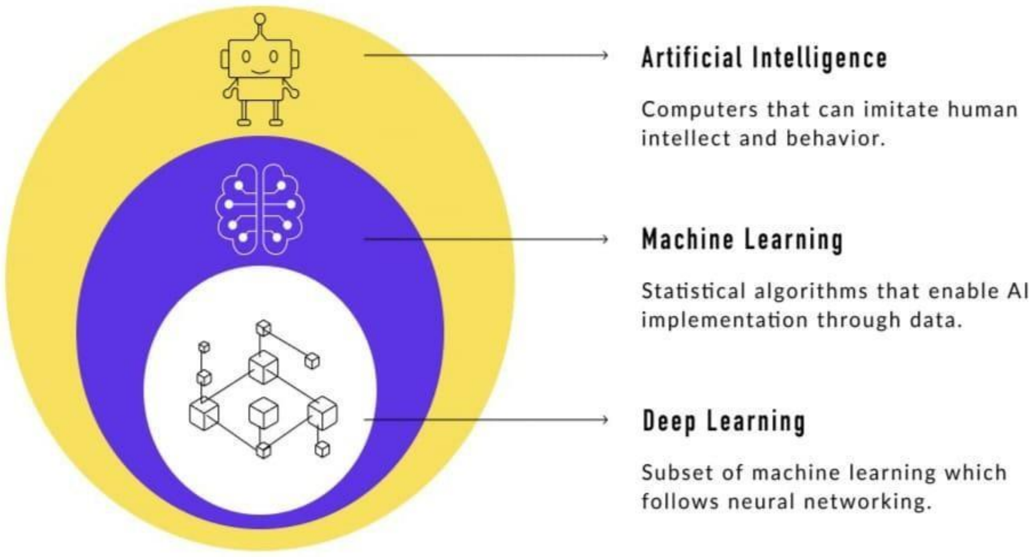

Artificial Intelligence
Computers that can imitate human intellect and behavior.
Machine Learning
Statistical algorithms that enable AI implementation through data.
Deep Learning
Subset of machine learning which follows neural networking.

Fig 1.1 Architecture of AI ML and DL

- Artificial Intelligence is the concept of creating smart intelligent machines.
- Machine Learning is a subset of artificial intelligence that helps you to build AI-driven applications
- Deep learning is a subset of machine learning that uses vast volumes of data and complex algorithms to train a model

1.2 Artificial Intelligence

Artificial intelligence, commonly referred to as AI, is the process of imparting data, information, and human intelligence to machines. The main goal of Artificial Intelligence is to develop self-reliant machines that can think and act like humans. These machines can mimic human behaviour and perform tasks by learning and problem-solving. Most of the AI systems simulate natural intelligence to solve complex problems. An example of AI-driven product is Amazon Echo.

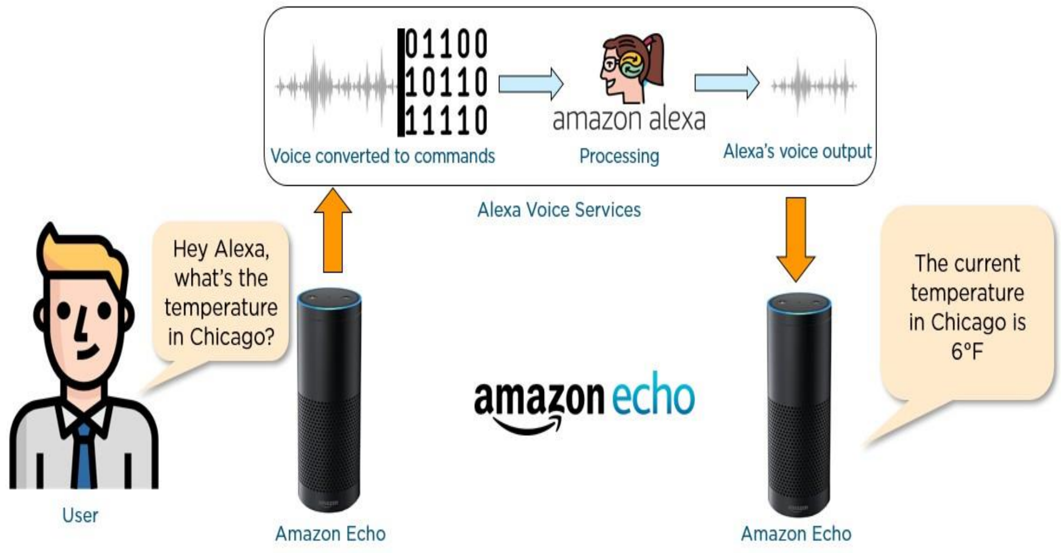


Fig 1.2 Example of AI

Amazon Echo is a smart speaker that uses Alexa, the virtual assistant AI technology developed by Amazon. Amazon Alexa is capable of voice interaction, playing music, setting alarms, playing audiobooks, and giving real-time informationsuch as news, weather, sports, and traffic reports. As we can see in the figure 1.2, the person wants to know the current temperature in Chicago. The person's voice is first converted into a machine-readable format. The formatted data is then fed into the Amazon Alexa system for processing and analysing. Finally, Alexa returns the desired voice output via Amazon Echo.

1.3 Machine Learning

Machine learning is a core sub-area of Artificial Intelligence (AI). ML applications learn from experience (or to be accurate, data) like humans do without direct programming. When exposed to new data, these applications learn, grow, change, and develop by themselves. In other words, machine learning involves computers finding insightful information without being told where to look. Instead, they do this by leveraging algorithms that learn from data in an iterative process. be considered to be "learning" from the data. Once these models have been fit to previously seen data, they can be used to predict and understand aspects of newly observed data.

1.3.1 Working of ML

Machine learning accesses vast amounts of data (both structured and unstructured) and learns from it to predict the future. It learns from the data by using multiple algorithms and techniques. Some popular machine learning techniques include decision trees, neural networks, support vector machines, and k- nearest neighbours. To train machine learning models, large amounts of data are needed, and the quality and diversity of the data play an important role in the accuracy and effectiveness of the models. Below figure 1.3 shows that how a machine learns from data.

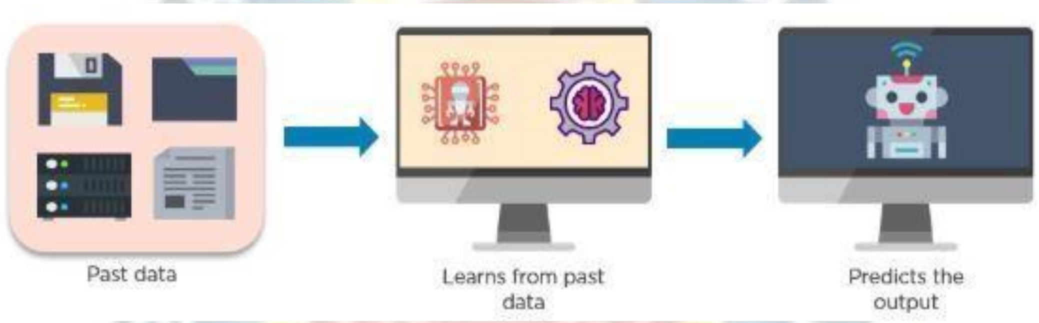


Fig 1.3 Working of ML

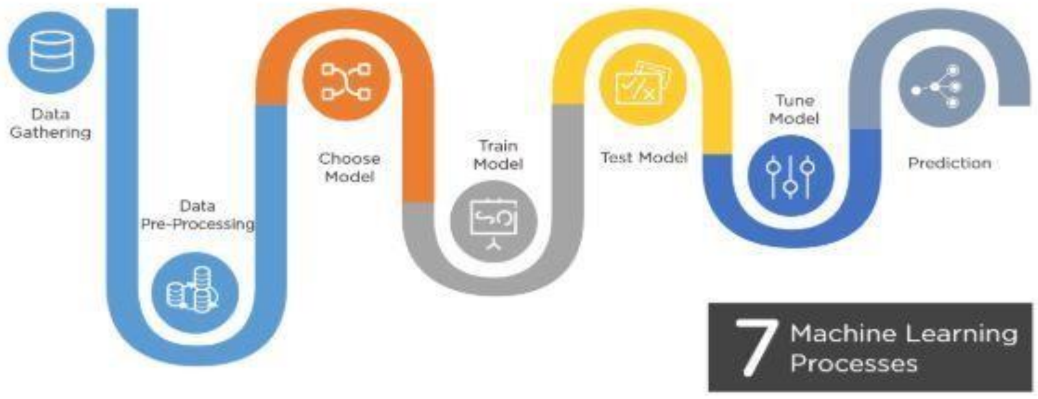


Fig 1.4 Processes of Machine Learning

1.3.2 Types of Machine Learning

Machine learning algorithms are classified into three main categories:

1. Supervised Learning
2. Unsupervised Learning
3. Reinforcement Learning

1.3.2.1 Supervised Learning

In supervised learning, the data is already labelled, which means you know the target variable. Using this method of learning, systems can predict future outcomes based on past data. It requires that at least an input and output variable be given to the model for it to be trained.
Consider an example of a supervised learning method. The algorithm is trained using labelled data of dogs and cats. The trained model predicts whether the new image is that of a cat or a dog as shown in the figure 1.5.

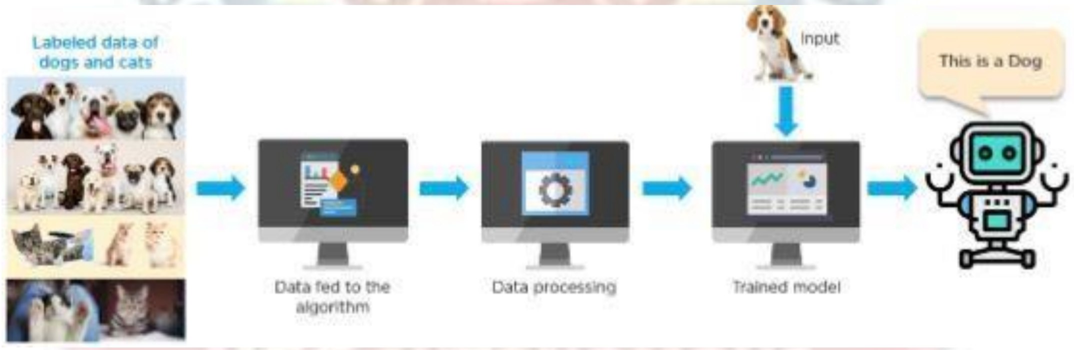


Fig 1.5 Example of Supervised Learning

Some examples of supervised learning include linear regression, logistic regression,support vector machines, Naive Bayes, and decision tree.

Supervised learning is further classified into two types they are-

1. Regression.
2. Classification

1.3.2.2 Unsupervised Learning

Unsupervised learning algorithms employ unlabeled data to discover patterns from the data on their own. The systems are able to identify hidden featuresfrom the input data provided. Once the data is more readable, the patterns and similarities become more evident.

Below figure 1.6 is an example of an unsupervised learning method that trains a model using unlabelled data. In this case, the data consists of different vehicles. The purpose of the model is to classify each kind of vehicle.

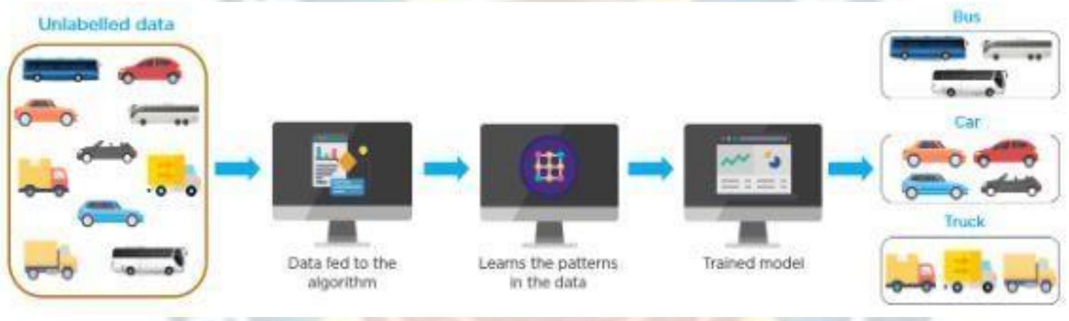


Fig 1.6 Example of unsupervised learning

Some examples of unsupervised learning include k-means clustering, hierarchicalclustering and anomaly detection.

1.3.2.3 Reinforcement Learning

The goal of reinforcement learning is to train an agent to complete a task within an uncertain environment as shown in the figure 1.7. The agent receives observations and a reward from the environment and sends actions to the environment. The reward measures how successful action is with respect to completing the task goal.

In reinforcement learning, developers devise a method of rewarding desired behaviours and punishing negative behaviours. This method assigns positive values to the desired actions to encourage the agent and negative values to undesired behaviours. This programs the agent to seek long-term and maximum overall reward to achieve an optimal solution.

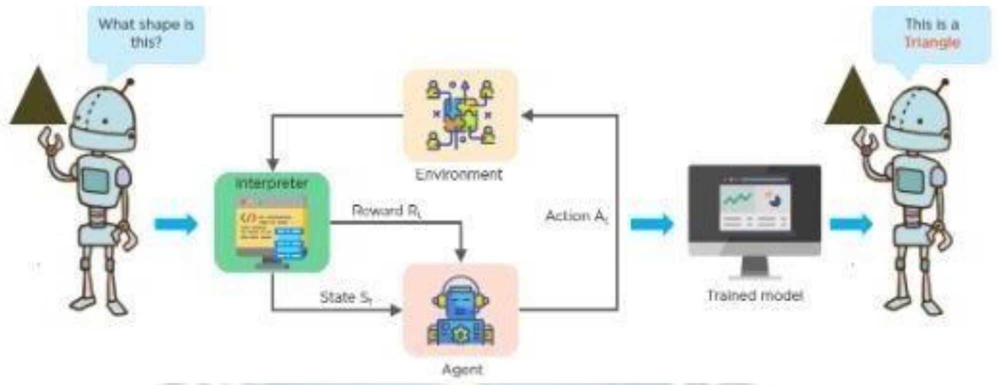


Fig 1.7 Example of reinforcement learning

1.3.3 Applications of ML

- Speech synthesis
- Weather forecasting and prediction
- Stock market analysis and forecasting
- Speech recognition
- Customer segmentation
- Object recognition
- Fraud detection and prevention
- Sales forecasting for different products
- Recommendation of products to customer in online shopping

1.3.4 Advantages of ML

- **Automatically:** Machine learning methods are automated processes (algorithms) that create algorithms. The methods run on data and produce a model that specifies how to achieve the program‘s goal.
- **Fast:** Machine learning methods save your time. The methods can analyse sample input data and deliver a program faster than you could manually write one.
- **Accurate:** Machine learning methods can do a better job than us. As automated methods, they can run longer on more data than us in order to make more accurate decisions.

- **Scale:** Machine learning methods can provide solutions to problems that we cannot solve.

1.3.5 Disadvantages of ML

1. Data Acquisition

Machine Learning requires massive data sets to train on, and these should be inclusive/unbiased, and of good quality. There can also be times where they must wait for new data to be generated.

2.Time and Resources

ML needs enough time to let the algorithms learn and develop enough tofulfil their purpose with a considerable amount of accuracy and relevancy. It also needs massive resources to function. This can mean additional requirements of computer power for you.

3.Interpretation of Results

Another major challenge is the ability to accurately interpret results generatedby the algorithms. You must also carefully choose the algorithms for your purpose.

4.High error-susceptibility

Machine Learning is autonomous but highly susceptible to errors. Suppose you train an algorithm with data sets small enough to not be inclusive. You end up with biased predictions coming from a biased training set. This leads to irrelevant advertisements being displayed to customers. In the case of ML, such blunders can set off a chain of errors that can go undetected for long periods of time. And when they do get noticed, it takes quite some time to recognize the source of the issue, and even longer to correct it.

1.4 Deep Learning

Deep learning is a subset of machine learning that deals with algorithms inspired by the structure and function of the human brain. Deep learning algorithms can work with an enormous amount of both structured and unstructured data. Deep learning‘s core concept lies in artificial neural networks, which enable machines to make decisions.

The major difference between deep learning vs. machine learning is the way data is presented to the machine. Machine learning algorithms usually require structured data, whereas deep learning networks work on multiple layersof artificial neural networks as shown in the figure 1.8.

Deep learning models are capable enough to focus on the accurate features themselves by requiring a little guidance from the programmer and are very helpful in solving out the problem of dimensionality. Deep learning algorithms are used, especially when we have a huge no of inputs and outputs.

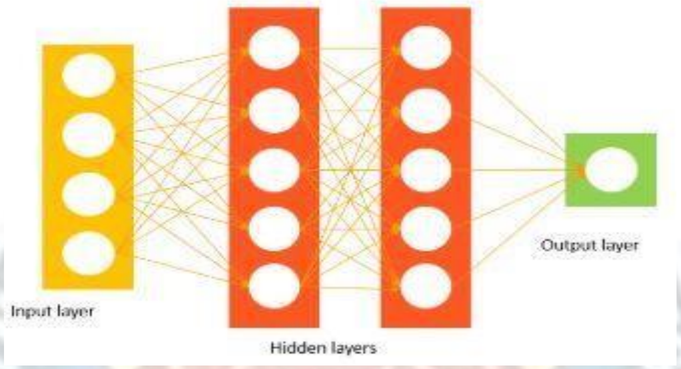


Fig 1.8 Simple neural network

The network has an input layer that accepts inputs from the data. The hidden layer is used to find any hidden features from the data. The output layer then provides the expected output. The term "deep" in deep learning refers to the fact that these networks are typically composed of many layers, allowing them to learn and represent increasingly complex patterns and relationships in data as shown in figure 1.9.

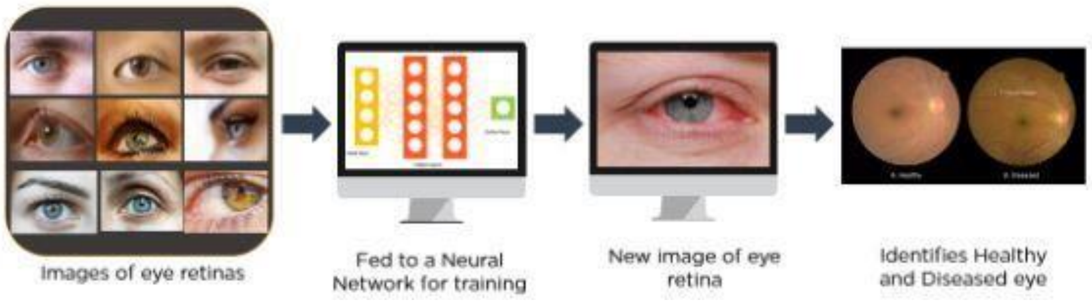

Images of eye retinas
Fed to a Neural Network for training
New image of eye retina
Identifies Healthy and Diseased eye

Fig 1.9 Example of deep learning

1.4.1 Working of Deep learning

1. Calculate the weighted sums.
2. The calculated sum of weights is passed as input to the activation function.
3. The activation function takes the ―weighted sum of input‖ as the input to the function, adds a bias, and decides whether the neuron should be fired or not.
4. The output layer gives the predicted output.
5. The model output is compared with the actual output. After training the neural network, the model uses the back propagation method to improve the performance of the network. The cost function helps to reduce the error rate.

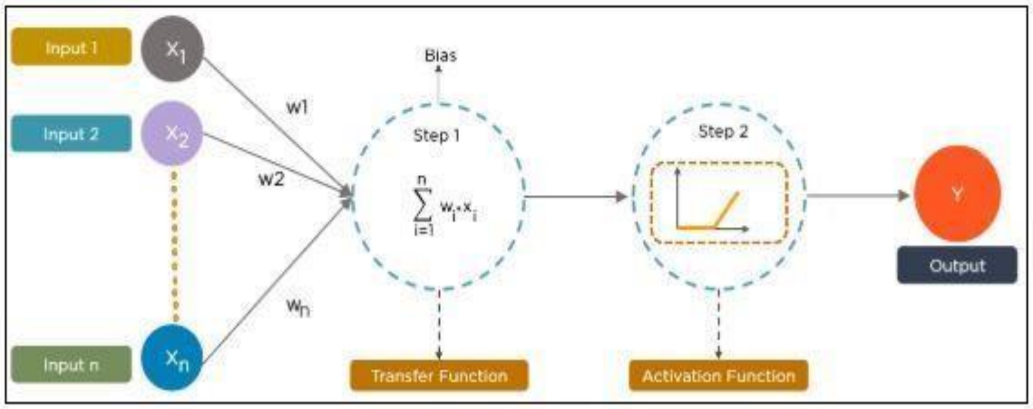


Fig 1.10 Working of Deep Learning

In addition to scalability, another often cited benefit of deep learning models is their ability to perform automatic feature extraction from raw data, also called feature learning as shown in the figure 1.10.

However, deep learning also has some limitations and challenges. One of the biggest challenges is the need for large amounts of labelled data to train the models effectively. Additionally, deep neural networks can be computationally expensive to train and require specialized hardware such as graphics processing units (GPUs) to speed up the process.

Despite these challenges, deep learning has had a major impact on many fields, and is likely to continue to be a key area of research and development in the coming years.

1.4.2 Advantages of DL

- Maximum utilization of unstructured data
- Elimination of the need for feature engineering
- Ability to deliver high quality results
- Elimination of unnecessary costs
- Elimination of the need of data labelling

1.4.3 Applications of DL

- Natural language processingPixel Restoration
- Fraud detection
- Health Care
- Self-Driving Cars
- Visual Recognition

1.4.4 Types of Deep Neural Networks

- **Convolutional Neural Network (CNN) -** CNN is a class of deep neural networks most commonly used for image analysis.
- **Recurrent Neural Network (RNN) -** RNN uses sequential information to build a model. It often works better for models that have to memorize past data.
- **Generative Adversarial Network (GAN) -** GAN are algorithmic architectures that use two neural networks to create new, synthetic instances of data that pass for real data. A GAN trained on photographs can generate new photographs that look at least superficially authentic to human observers.
- **Deep Belief Network (DBN) -** DBN is a generative graphical model that is composed of multiple layers of latent variables called hidden units. Each layer is interconnected, but the units are not.

CHAPTER 2
LITERATURE SURVEY

2.1 Survey on Evolving Deep Learning Neural Network Architectures

AUTHORS: Abul Bashar

The deep learning being a subcategory of the machine learning follows the human instincts of learning by example to produce accurate results. The deep learning performs training to the computer frame work to directly classify the tasks from the documents available either in the form of the text, image, or the sound. Most often the deep learning utilizes the neural network to perform the accurate classification and is referred as the deep neural networks; one of the most common deep neural networks used in a broader range of applications is the convolution neural network that provides an automated way of feature extraction by learning the features directly from the images or the text unlike the machine learning that extracts the features manually. This enables the deep learning neural networks to have a state of art accuracy that mostly expels even the human performance. So the paper is to present the survey on the deep learning neural network architectures utilized in various applications for having an accurate classification with an automated feature extraction.

2.2 A Novel Online Dynamic Temporal Context Neural Network Framework for the Prediction of Road Traffic Flow

AUTHORS: Zoe Bartlett et al

Traffic flow exhibits different magnitudes of temporal patterns, such as short-term (daily and weekly) and long-term (monthly and yearly). Existing research into road traffic flow prediction has focused on short-term patterns; little research has been done to determine the effect of different long-term patterns on road traffic flow prediction. Providing more temporal contextual information through the use of different temporal data segments could improve prediction results. In this paper, we have investigated different magnitudes of temporal patterns, such as short-term and long-term, through the use of different temporal data segments to understand how contextual temporal data can improve prediction. Furthermore, to learn temporal patterns dynamically, we have proposed a novel online dynamic temporal context neural network framework.

The framework uses different temporal data segments as input features, and during online learning, the updating scheme dynamically determines how useful a temporal data segment (short and long-term temporal patterns) is for prediction, and weights it accordingly for use in the regression model. Therefore, the framework can include short-term and relevant long-term patterns in the regression model leading to improved prediction results. We have conducted a thorough experimental evaluation with a real dataset containing daily, weekly, monthly and yearly data segments. The experiment results show that both short and long-term temporal patterns improved prediction accuracy. In addition, the proposed online dynamical framework improved predication results by 10.8% when compared with a deep gated recurrent unit model.

2.3 Detection of unwanted traffic congestion based on existing surveillance system using in freeway via a CNN architecture trafficNet

AUTHORS: P. Wang, L. Li, Y. Jin, and G. Wang

Detection of traffic congestion is important for route guidance using in intelligent transport system (ITS) to prevent jam escalation. Although the surveillance system has been used in freeway for years, it is hard to automatically identify and report traffic congestion in complicated transportation scene according to various illumination, weather and other disturbances. The detection process based on human eye is time-consuming and tedious as the machine detection accuracy is not high enough to meet the requirements of practical applications.

In this paper, a new classifier is proposed using convolutional neural networks (CNN) to generate four TrafficNet based on two championships of ILSVRC including AlexNet and VGGNet. Instead of using fully-connected layers in AlexNet and VGGNet, a support vector machine (SVM) are used after CNN architecture. Congestion and non-congestion images are trained and tested through this new structure. Image database with more than 30000 images are extracted from existing traffic surveillance video and corresponding labels are added manually. With database, those TrafficNet are trained and tested, detection accuracy and training time of those TrafficNet are compared. The experimental results show that the accuracy of proposed method can reach up to 90%, which is much higher than traditional method based on feature extraction without deep learning.

2.4 LSTM network: A deep learning approach for short-term traffic forecast

AUTHORS: Weihai Chen, Zheng Zhao, Jingmeng Liu and Peter C. Y. Chen

Short-term traffic forecast is one of the essential issues in intelligent transportation system. Accurate forecast result enables commuters make appropriate travel modes, travel routes, and departure time, which is meaningful in traffic management. To promote the forecast accuracy, a feasible way is to develop a more effective approach for traffic data analysis. The availability of abundant traffic data and computation power emerge in recent years, which motivates us to improve the accuracy of short-term traffic forecast via deep learning approaches.

A novel traffic forecast model based on long short-term memory (LSTM) network is proposed. Different from conventional forecast models, the proposed LSTM network considers temporal-spatial correlation in traffic system via a two-dimensional network which is composed of many memory units. A comparison with other representative forecast models validates that the proposed LSTM network can achieve a better performance.

2.5 Deep visual tracking: Review and experimental comparison

AUTHORS: Peixia Li, Dong Wang, Lijun Wang and Huchuan Lu

Recently, deep learning has achieved great success in visual tracking. The goal of this paper is to review the state-of-the-art tracking methods based on deep learning. First, we introduce the background of deep visual tracking, including the fundamental concepts of visual tracking and related deep learning algorithms.

Second, we categorize the existing deep-learning-based trackers into three classes according to network structure, network function and network training.

For each categorize, we explain its analysis of the network perspective and analyze papers in different categories. Then, we conduct extensive experiments to compare the representative methods on the popular OTB-100, TC-128 and VOT2015 benchmarks. Based on our observations, we conclude that:

(1) The usage of the convolutional neural network (CNN) model could significantly improve the tracking performance.

(2) The trackers using the convolutional neural network (CNN) model to distinguish the tracked object from its surrounding background could get more accurate results, while using the CNN model for template matching is usually faster.

(3) The trackers with deep features perform much better than those with low-level hand-crafted features.

(4) Deep features from different convolutional layers have different characteristics and the effective combination of them usually results in a more robust tracker.

(5) The deep visual trackers using end-to-end networks usually perform better than the trackers merely using feature extraction networks.

(6) For visual tracking, the most suitable network training method is to per-train networks with video information and online fine-tune them with subsequent observations. Finally, we summarize our manuscript and highlight our insights, and point out the further trends for deep visual tracking.

CHAPTER 3

SYSTEM ANALYSIS AND DESIGN

3.1 Existing Design

Existing system uses **K-Nearest Neighbor(KNN**) methodology which involves preprocessing and training the dataset.

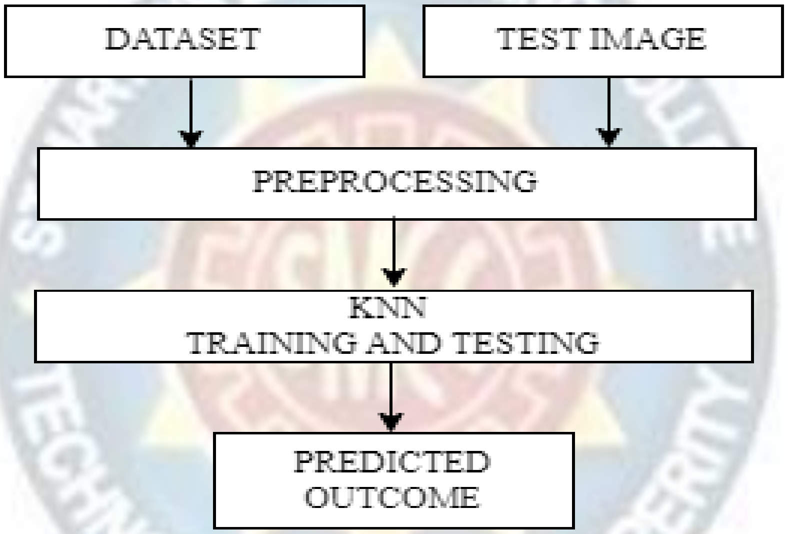


Fig 3.1 Block Diagram Representation of Existing System

Dataset : A machine learning dataset is a collection of data that is used to train the model. A dataset acts as an example to teach the machine learning algorithm how to make predictions. The common types of data include: Text data. Image data.

Pre-Processing : Preprocessing refers to all the transformations on the raw data before it is fed to the machine learning or deep learning algorithm. It is common for both proposed and existing system.

The main 3 steps involved in Preprocessing are:

1) Size of the image is normalized
2) Image data types are converted into double
3) Image to vector conversion (image pixel reading)

3.1.1 KNN Testing and Training:

K-Nearest Neighbour is one of the simplest Machine Learning algorithms based on Supervised Learning technique.
K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.
K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm.

Need for K-NN Algorithm:

Suppose there are two categories, i.e., Category A and Category B, and we have a new data point x1, so this data point will lie in which of these categories. To solve this type of problem, we need a K-NN algorithm. With the help of K-NN, we can easily identify the category or class of a particular dataset.

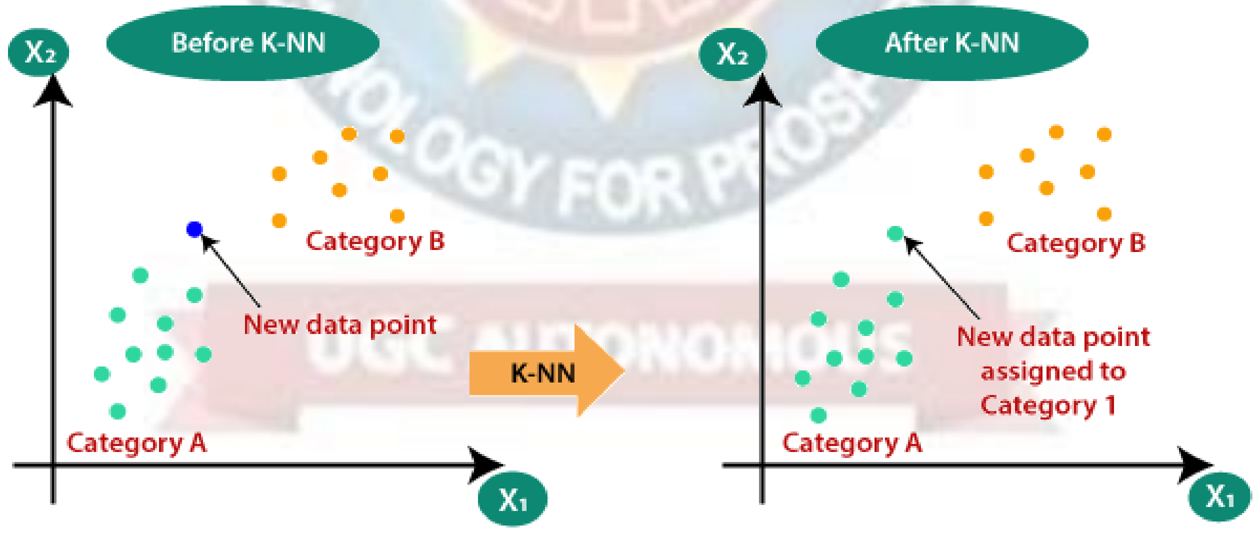


Fig 3.1.1 Before KNN **Fig 3.1.2 After KNN**

3.1.2 KNN Working

The K-NN working can be explained on the basis of the below algorithm:

- **Step-1:** Select the number K of the neighbors
- **Step-2:** Calculate the Euclidean distance of **K number of neighbors**
- **Step-3:** Take the K nearest neighbors as per the calculated Euclidean distance.
- **Step-4:** Among these k neighbors, count the number of the data points in each category.
- **Step-5:** Assign the new data points to that category for which the number of the neighbor is maximum.
- **Step-6:** Our model is ready.

Example: Suppose we have a new data point and we need to put it in the required category. Consider the below image:

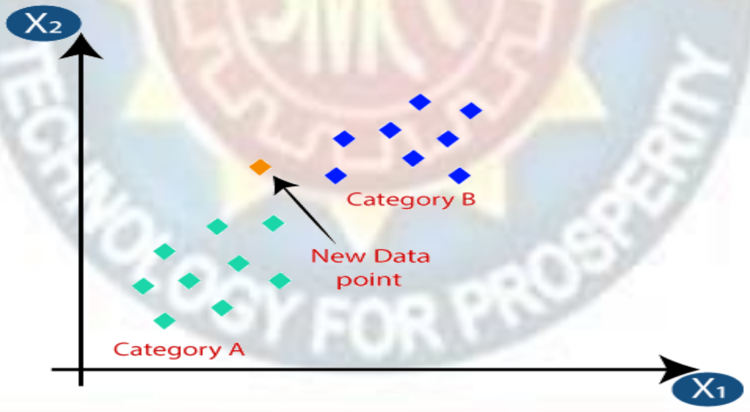


Fig 3.1.3 New Data Point

- Firstly, we will choose the number of neighbors, so we will choose the k=5.
- Next, we will calculate the **Euclidean distance** between the data points. The Euclidean distance is the distance between two points, which we have already studied in geometry. It can be calculated as:

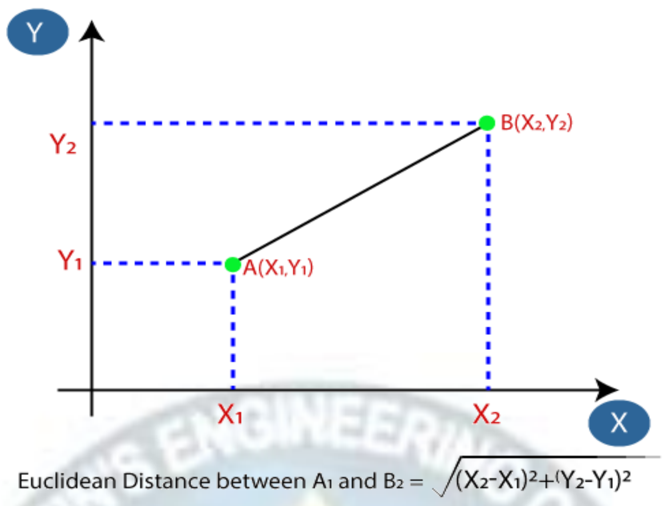


Fig 3.1.4 Calculation of Euclidean Distance

- By calculating the Euclidean distance we got the nearest neighbors, as three nearest neighbors in category A and two nearest neighbors in category B. Consider the below image:

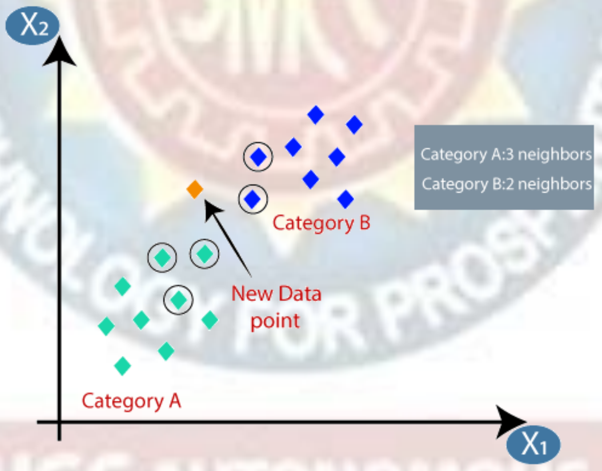


Fig 3.1.5 : Classification of New Data Point is Performed

- As we can see the 3 nearest neighbors are from category A, hence this new data point must belong to category A.

3.1.3 Disadvantages of Existing System

- Always needs to determine the value of K which may be complex some time.
- The computation cost is high because of calculating the distance between the data points for all the training samples.
- High error rates
- Low accuracy
- Cannot applicable to large scale dataset

3.2 Proposed system

The traffic surveillance system is accumulated with an enormous amount of data regarding road traffic each and every second. Monitoring these data with the human eye is a tedious process and it also requires manpower for monitoring. Deep learning approach (Convolutional Neural Network) can be utilized for traffic monitoring and control. The traffic surveillance data are pre-processed to construct the training dataset. The Traffic net is constructed by transferring the network to traffic applications and retraining it with self-established data set. This Traffic net can be used for regional detection in large scale applications. Further, it can be implemented across-the-board. The efficiency is admirably verified through speedy discovery in the high accuracy in the case study. The tentative assessment could pull out to its successful application to a traffic surveillance system and has potential enrichment for the intelligent transport system in future.

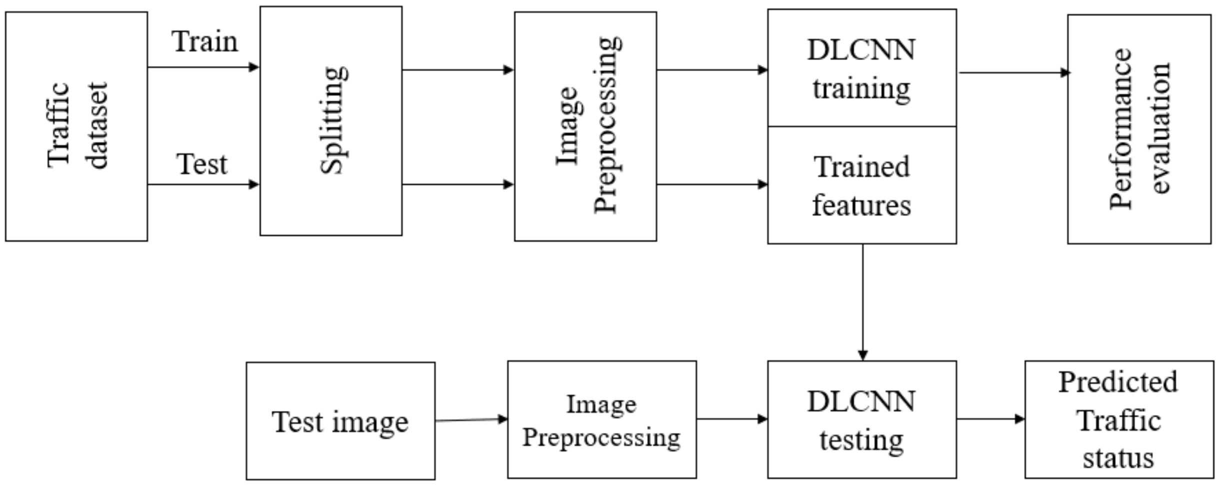


Fig 3.2 Proposed Method

Figure 3.2 shows the block diagram of proposed method. Initially, TrafficNet dataset is spitted into 80% for training and 20% for testing. Then, dataset preprocessing operation is performed to normalize the entire dataset. The image preprocessing operation converts the all the images into uniform size. Further, DLCNN is used for prediction of traffic status i.e., dense traffic, low traffic, accident, and fire occurred from test sample. The performance evaluation is carried out to show supremacy of proposed method.

The proposed system overcomes the drawbacks of the existing system by applying the deep learning techniques which is the present trending technique, since it is a multilayer neural network. The deep neural network is effective for recognition. The interest to use hidden layers has surpassed traditional techniques, especially in pattern recognition as in figure 3.2. One of the most popular deep neural networks is Convolutional Neural Networks. In simple words, ConvNet isto reduce the images into a form that is easier to process, without losing features that are critical for getting a good prediction. A convolution neural network has multiple hidden layers that help in extracting information from an image and it is a kind of network architecture for deep learning algorithms and is specially used for image recognition and tasks that involve the processing of pixel data.

3.3 Applications

- It can be applicable to metropolitan areas widely.
- It can be applicable to all the places where cameras are existed.
- Can be very much useful in monitoring the traffic and also helps in knowing the conditions of traffic.
- If embedded is included with this project then it can send messages to the corresponding departments like hospitals, fire departments etc.,

CHAPTER 4

IMPLEMENTATION

The implementation of project involves major steps such as:

- Getting the dataset
- Importing libraries
- Importing datasets
- Finding Missing Data
- Encoding Categorical Data
- Splitting dataset into training and test set
- Feature scaling

4.1 Traffic condition dataset

The input dataset of dissimilar classes is gathered from the net. The assessment of output class is set next to the obtained dataset. Four folders namely sparse_traffic, dense_traffic, fire, accident, every folder contains 900 images are generated for training and validation purposes. The folder name represents the class value for classifying output.

4.2 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 amachine learning model. When creating a project, it is not always a case that we come across the clean and formatted data a real-world data generally contains noises, missing values, and maybe in an unusable format which cannot be directly used for machine learning models 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 and it also increases the accuracy and efficiency of a machine learning model. The main 3 steps involved in Pre-processing are:

i) Size of the image is normalized

ii) Image data types are converted into double/float

iii) Image to vector conversion (image pixel reading)

4.2.1 Splitting the Dataset into the Training set and Test set:

In machine learning data pre-processing, we divide our dataset into a training set and test set. This is one of the crucial steps of data pre-processing as by doing this; we can enhance the performance of our machine learning model.

Suppose if we have given training to our machine learning model by a dataset and we test it by a completely different dataset. Then, it will create difficulties for our model to understand the correlations between the models.

If we train our model very well and its training accuracy is also very high, but we provide a new dataset to it, then it will decrease the performance. So we always try to make a machine learning model which performs well with the trainingset and also with the test dataset as shown in figure 4.1. Here, we can define these datasets as:

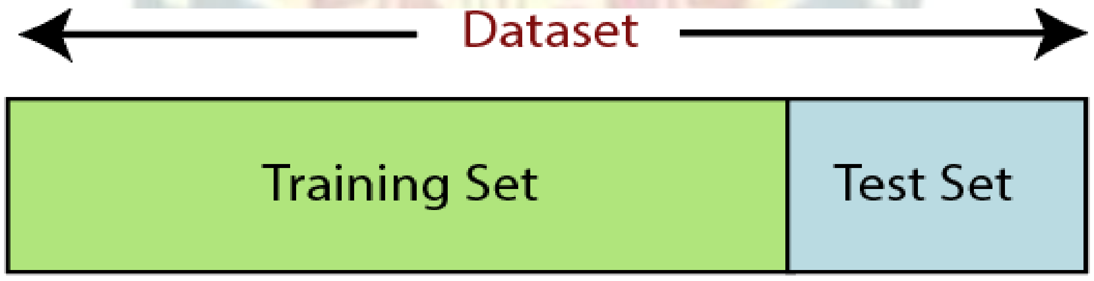


Fig 4.1 Training and Testing data

Training Set: A subset of dataset to train the machine learning model, and wealready know the output.

Test set: A subset of dataset to test the machine learning model, and by using thetest set, model predicts the output.

4.3 Custom CNN

A convolutional neural network (CNN) is a type of Artificial Neural Network(ANN) used in image recognition and processing which is specially designed for processing data(pixels). The first thing we should do is feed the pixelsof the image in the form of arrays to the input layer of the neural network (MLP networks used to classify such things).

Custom CNNs, or custom convolutional neural networks, are neural networks that are designed and trained for specific tasks. A custom CNN can be created by defining the network architecture, which involves specifying the number of layers, the type of layers, and their connections shown in figure 4.2.

According to the facts, training and testing of any deep neural network or transfer learning involves in allowing every source data via a succession of convolution layers by, rectified linear unit (ReLU), max pooling, fully connected layer and utilize SoftMax layer with classification layer to categorize the objects with probabilistic values ranging from ,0,1-.

Convolution layer as is the primary layer to extract the features from a source image and maintains the relationship between pixels by learning the features of image by employing tiny blocks of source data. It‘s a mathematical function which considers two inputs like source image $I(x, y, d)$ where x and y denotes the spatial coordinates i.e., number of rows and columns. d is denoted as dimension of an image (here $d = 3$, since the source image is RGB) and a filter or kernel with similar size of input image and can be denoted as $F(k_x, k_y, d)$.

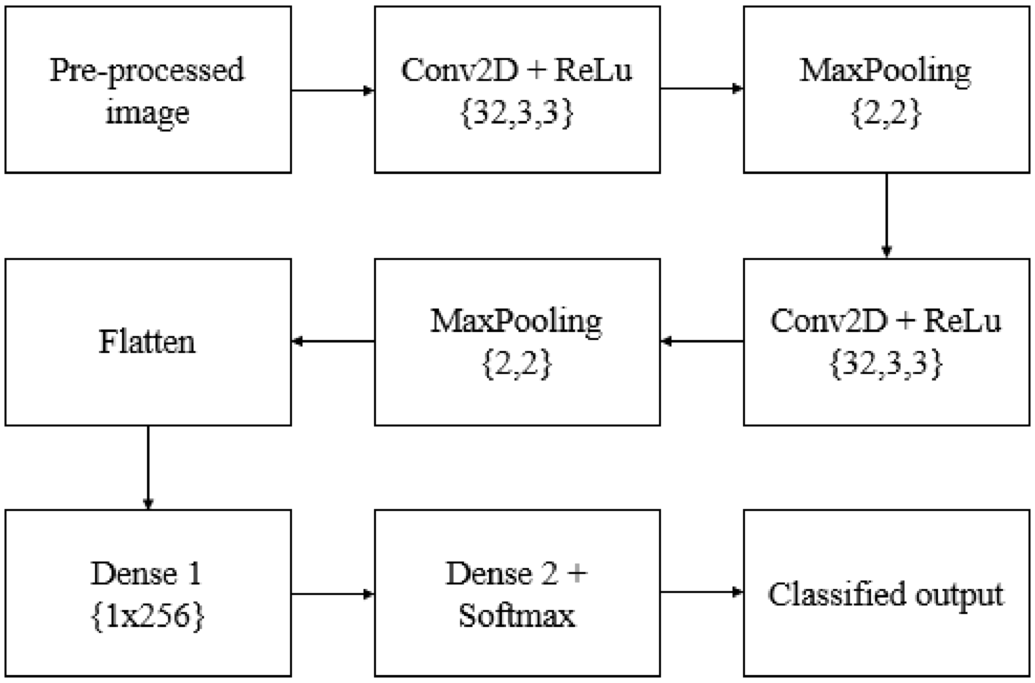


Fig 4.2 CNN Architecture

Convolutional neural networks are generally composed of three parts. Convolution layer for feature extraction. The convergence layer, also known as the pooling layer, is mainly used for feature selection. The number of parameters is reduced by reducing the number of features. The full connection layer carries out the summary and output of the characteristics. A convolution layer is consisting of a convolution process and a nonlinear activation function ReLU. A typical architecture of CNN model for crop Traffic condition recognition is shown in Figure 2.

The leftmost image is the input layer, which the computer understands as the input of several matrices. Next is the convolution layer, the activation function of which uses ReLU. The pooling layer has no activation function. The combination of convolution and pooling layers can be constructed many times. The combination of convolution layer and convolution layer or convolution layer and pool layer can be very flexibly, which is not limited when constructing the model. But the most common CNN is a combination of several convolution layers and pooling layers. Finally, there is a full connection layer, which acts as a classifier and maps the learned feature representation to the sample label space.

Convolutional neural network mainly solves the following two problems:

1) Problem of too many parameters: It is assumed that the size of the input picture is $50 * 50 * 3$. If placed in a fully connected feedforward network, there are 7500 mutually independent links to the hidden layer. And each link also corresponds to its unique weight parameter. With the increase of the number of layers, the size of the parameters also increases significantly. On the one hand, it will easily lead to the occurrence of over-fitting phenomenon. On the other hand, the neural network is too complex, which will seriously affect the training efficiency. In convolutional neural networks, the parameter sharing mechanism makes the same parameters used in multiple functions of a model, and each element of the convolutional kernel will act on a specific position of each local input. The neural network only needs to learn a set of parameters and does not need to optimize learning for each parameter of each position.

2) Image stability: Image stability is the local invariant feature, which means that the natural image will not be affected by the scaling, translation, and rotation of the image size. Because in deep learning, data enhancement is generally needed to improve performance, and fully connected feedforward neural is difficult to ensure the local invariance of the image. This problem can be solved by convolution operation in convolutional neural network.

4.3.1 Convolution layer

According to the facts, training and testing of DLCNN involves in allowing every source image via a succession of convolution layers by a kernel or filter, rectified linear unit (ReLU), max pooling, fully connected layer and utilize SoftMax layer with classification layer to categorize the objects with probabilistic values ranging from [0,1].

Convolution layer as depicted in Figure 4.3 is the primary layer to extract the features from a source image and maintains the relationship between pixels by learning the features of image by employing tiny blocks of source data. It's a mathematical function which considers two inputs like source image $I(x, y, d)$ where x and y denotes the spatial coordinates i.e., number of rows and columns. d is denoted as dimension of an image (here $d = 3$, since the source image is RGB) and a filter or kernel with similar size of input image and can be denoted as $F(k_x, k_y, d)$.

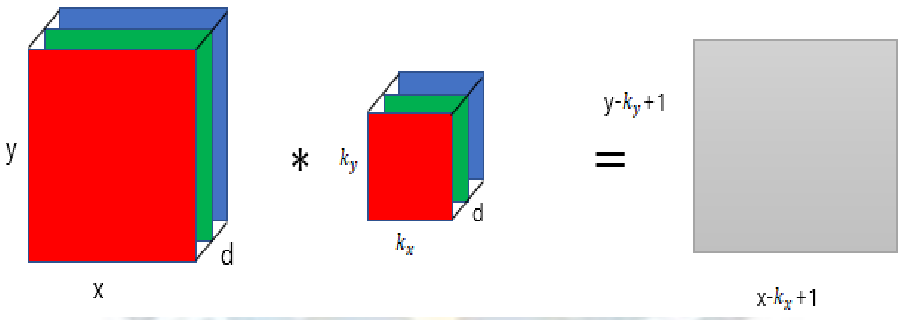


Fig 4.3 Representation of convolution layer process.

The output obtained from convolution process of input image and filter hasa size of C ₋$(x - k_x + 1), (y - k_y + 1), 1/$, which is referred as feature map. Let us assume an input image with a size of 5×5 and the filter having the size of 3×3. The feature map of input image is obtained by multiplying the input image values with the filter values shown in figure 4.4.

$$\underbrace{\begin{array}{|c|c|c|c|c|}\hline 1 & 1 & 1 & 0 & 0 \\ \hline 0 & 0 & 1 & 1 & 1 \\ \hline 1 & 1 & 0 & 0 & 1 \\ \hline 0 & 0 & 0 & 1 & 1 \\ \hline 1 & 1 & 1 & 0 & 0 \\ \hline \end{array}}_{\text{5x5 image}} * \underbrace{\begin{array}{|c|c|c|}\hline 1 & 0 & 1 \\ \hline 0 & 1 & 0 \\ \hline 1 & 0 & 1 \\ \hline \end{array}}_{\text{3x3 kernel}} = \underbrace{\begin{array}{|c|c|c|}\hline 3 & 3 & 3 \\ \hline 2 & 2 & 3 \\ \hline 3 & 2 & 3 \\ \hline \end{array}}_{\text{Feature map}}$$

Fig: 4.4 An image with size 5×5 is convolving with 3×3 kernel and Convolved feature map.

- The process of convolution involves a kernel or filter inside this layer moving across the receptive fields of the image, checking if a feature is present in the image as shown in figure 4.5.
- Over multiple iterations, the kernel sweeps over the entire image. After each iteration a dot product is calculated between the input pixels and the filter. The final output from the series of dots is known as a feature map or convolved feature. Ultimately, the image is converted into numerical values in this layer, which allows the CNN to interpret the image and extract relevant patterns from it.

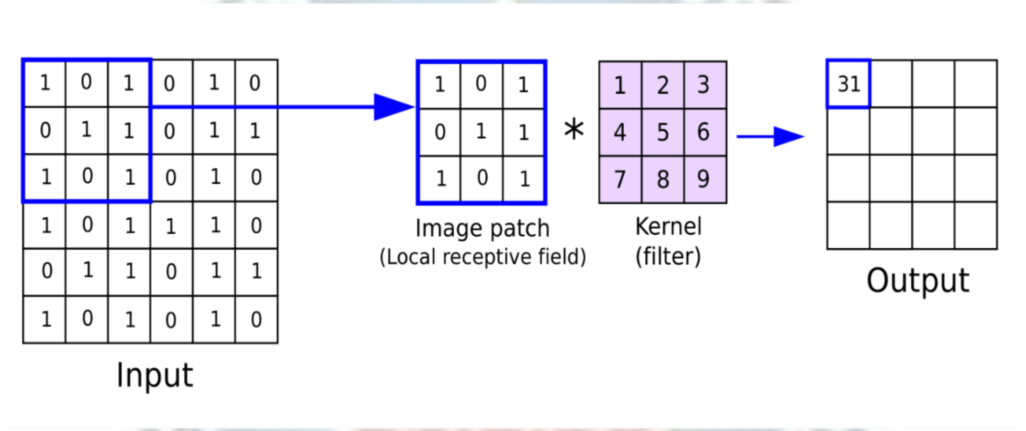


Fig 4.5 Representation of convolution layer process.

When designing a custom CNN, it is important to consider the input size, the output size, and the complexity of the task. The input size determines the size of thefirst layer of the network, while the output size determines the size of the last layer. The complexity of the task determines the number and type of layers in between. Learning a single filter specific to a machine learning task is a powerfultechnique. Yet, convolutional neural networks achieve much more in practice.

4.3.2 Multiple Filters

Convolutional neural networks do not learn a single filter; they, in fact, learn multiple features in parallel for a given input. For example, it is common for a convolutional layer to learn from 32 to 512 filters in parallel for a given input. This gives the model 32, or even 512, different ways of extracting features from an input, or many different ways of both ―learning to see‖ and after training, many different ways of ―seeing‖ the input data. This diversity allows specialization, e.g. not just lines, but the specific lines seen in your specific training data.

4.3.3 Multiple Channels

Colour images have multiple channels, typically one for each colour channel, such as red, green, and blue. From a data perspective, that means that a single image provided as input to the model is, in fact, three images. A filter must always have the same number of channels as the input, often referred to as –depth–. If an input image has 3 channels (e.g. a depth of 3), then a filter applied to that image must also have 3 channels (e.g. a depth of 3). In this case, a 3×3 filter would in fact be 3x3x3 or [3, 3, 3] for rows, columns, and depth. Regardless of the depth of the input and depth of the filter, the filter is applied to the input using a dot product operation which results in a single value.

This means that if a convolutional layer has 32 filters, these 32 filters are not just two-dimensional for the two-dimensional image input, but are also three- dimensional, having specific filter weights for each of the three channels. Yet, each filter results in a single feature map. This means that the depth of the output of applying the convolutional layer with 32 filters is 32 for the 32 feature maps created.

4.3.4 Multiple layers

Convolutional layers are not only applied to input data, e.g. raw pixel values, but they can also be applied to the output of other layers. The stacking of convolutional layers allows a hierarchical decomposition of the input. Consider thatthe filters that operate directly on the raw pixel values will learn to extract low-levelfeatures, such as lines. The filters that operate on the output of the first line layers may extract features that are combinations of lower-level features, such as features that comprise multiple lines to express shapes. This process continues until very deep layers are extracting faces, animals, houses, and so on.

4.4 ReLU Layer

Networks those utilizes the rectifier operation for the hidden layers are cited as rectified linear unit (ReLU). This ReLU function $\mathcal{G}(\cdot)$ is a simple computation that returns the value given as input directly if the value of input is greater than zero elsereturns zero. This can be represented as mathematically using the function $\max(\cdot)$ over the set of 0 and the input 7 as follows:

$$\mathcal{G}(x) = \max\{0, x\}$$

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.

A ReLU layer performs:

- Element wise operation
- Has an output i.e. rectified feature map

In addition to its computational efficiency, ReLU has also been shown to improve the accuracy of neural networks for a variety of tasks, including image classification and natural language processing as shown in figure 4.6.

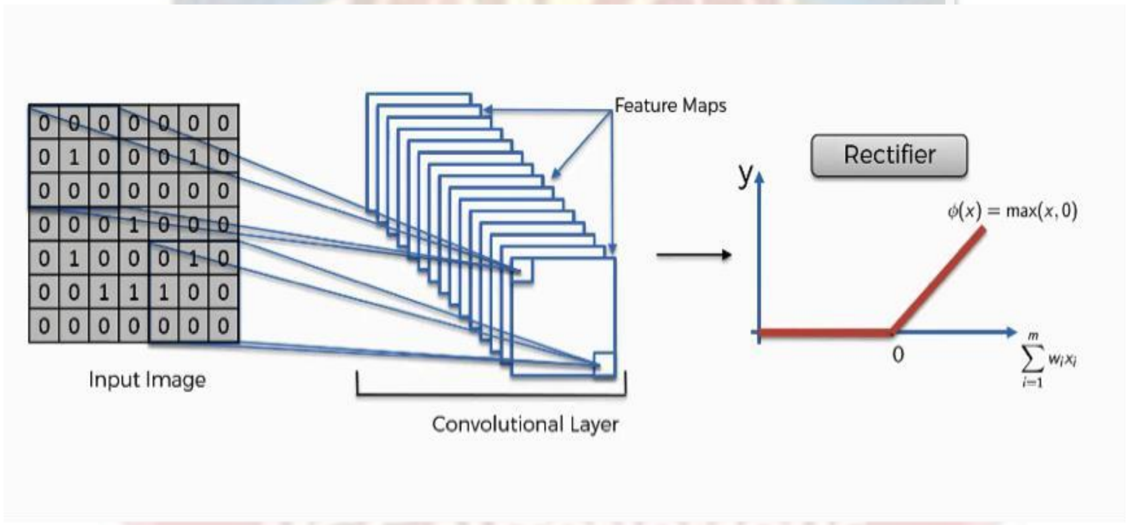


Fig 4.6 ReLU Layer

By removing negative values from the neurons' input signals, the rectifier function is effectively removing black pixels from the image and replacing them with gray pixels.

4.5 Max-pooling Layer

Now that the rectifier function has removed black pixels from our image, it's time to implement some maximum pooling techniques. The purpose of max poolingit to teach the convolutional neural networks to detect features in an image whenthe feature is presented in any manner. Pooling is the process that allows us to introduce spatial variance. There are numerous types of pooling (including sum pooling and mean pooling) but we will be working with max pooling.

Pooling is used to transform a feature map into a pooled feature map, which is smaller and is calculated based on the original feature map using a similar matrix overlay technique as shown figure 4.7.

Consider an example that transforms a 5x5 feature map into a pooled feature map of dimensions 3x3 using max pooling. We'll do this using a 2x2 overlay box. To start, place your 2x2 overlay box in the top-left corner of the feature map. Take the largest value contained in the matrix overlay - this becomes the first value inthe pooled feature map. Now move the box by the amount of your stride. Continue this process until the pooled feature map is full.

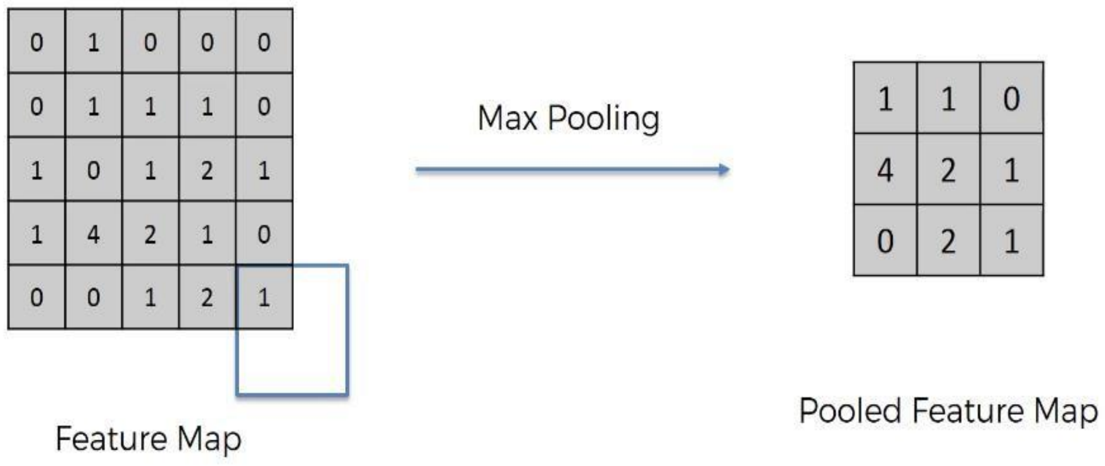


Fig 4.7 Max-pooling layer

Pooling is beneficial because it reduces the dimensionality of the data we're training on, which lowers the chance that our model will be overfitted on our training data. As well, it reduces the computational cost by reducing the number of parameters to learn and provides basic translation invariance to the internal representation. Max pooling is done by applying a max filter to (usually) non- overlapping sub regions of the initial representation.

4.6 SoftMax classifier

Generally, softmax function is added at the end of the output as shown in Figure 5, since it is the place where the nodes are meet finally and thus, they can be classified. Here, X is the input of all the models and the layers between X and Y are the hidden layers and the data is passed from

X to all the layers and Received by Y. Suppose, we have 10 classes, and we predict for which class the given input belongs to. So, for this what we do is allot each class with a particular predicted output. Which means that we have 10 outputs corresponding to 10 different class and predict the class by the highest probability it has.

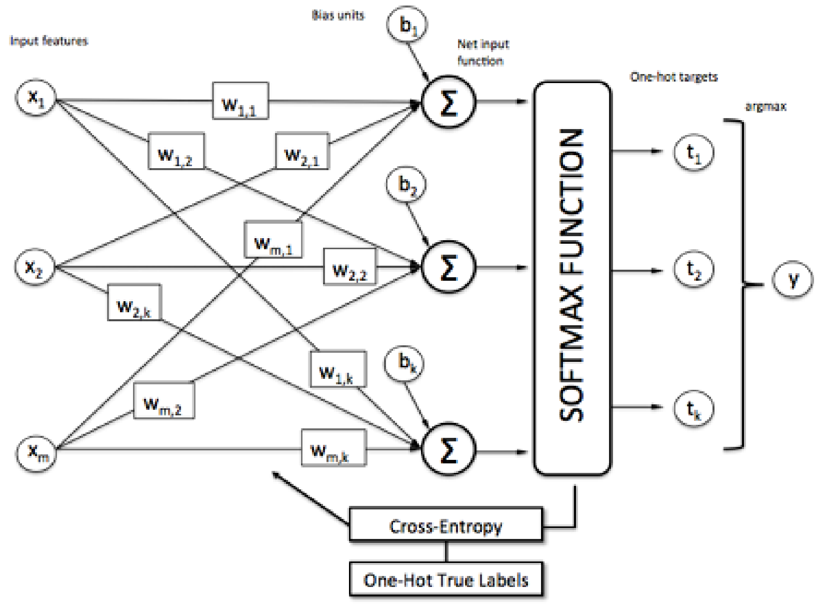


Fig 4.8 Traffic condition prediction using SoftMax classifier.

Softmax is an activation function that scales numbers/logits intoprobabilities. The output of a Softmax is a vector (say v) with probabilities of each possible outcome. The probabilities in vector v sums to one for all possible outcomes or classes as shown in the figure 4.8.

The sigmoid activation function can also be used as an activation function for multi-class classification problems where classes are non-mutually exclusive. Theseare often referred to as a multi-label classification rather than multi-class classification. The sigmoid activation function is not appropriate for multi-class classification problems with mutually exclusive classes where a multinomial probability distribution is required.

Instead, an alternate activation is required called the softmax function. This canbe achieved by calculating the exponent of each value in the list and dividing it by the sum of the exponent values.

Probability = exp (value) / sum v in list exp (v)

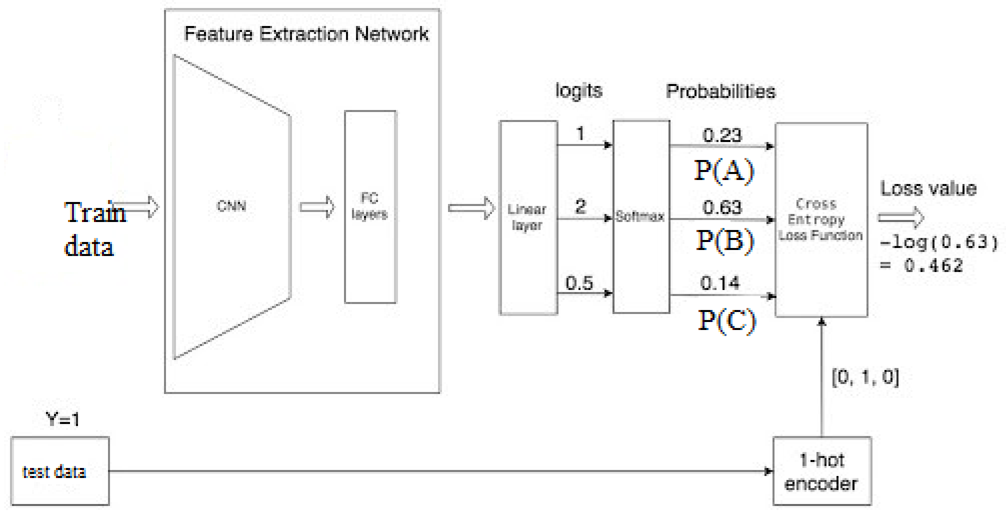


Fig 4.9 Example of softmax classifier with test data

In Figure 4.9, we must predict what the object that is present in the picture is.This is the place where softmax comes in handy.

As the model is already trained on some data. So, as soon as the picture is given, the model processes the pictures, send it to the hidden layers and then finally send to softmax for classifying the picture.

From the diagram, we see that the predictions are occurred. But generally, we don‘t know the predictions. But the machine must choose the correct predicted object. So, for machine to identify an object correctly, it uses a function called cross-entropy function.So, we choose more similar value by using the below cross-entropy formula.

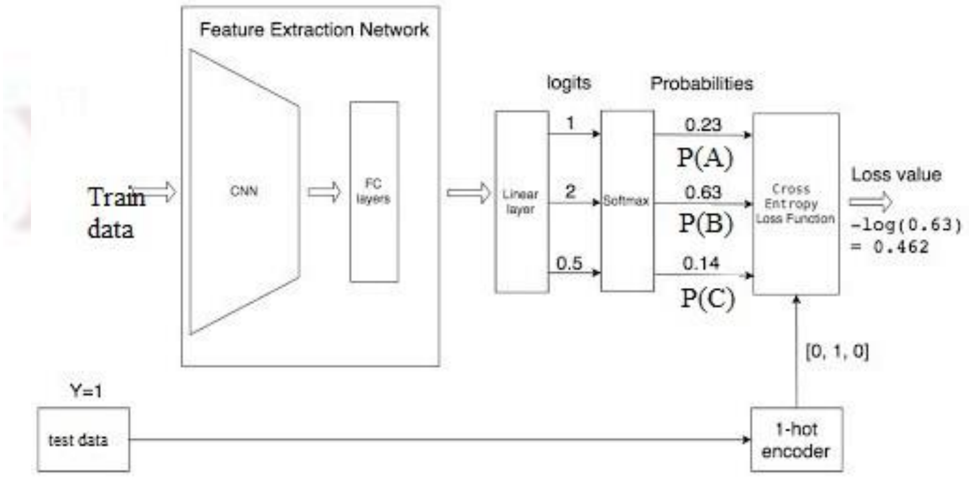


Fig 4.10 Example of softmax classifier with train data

In the above figure 4.9 we see that 0.462 is the loss of the function for class specific classifier. In the same way, we find loss for remaining classifiers. The lowest the loss function, the better the prediction is. The mathematical representation for loss function can be represented as: -

$$\mathrm{LOSS} = \mathrm{np.sum}(-\mathrm{Y} \; * \; \mathrm{np.log}(\mathrm{Y\_pred})$$

4.7 Advantages of Proposed system

- CNNs do not require human supervision for the task of identifyingimportant features.
- They are very accurate at image recognition and classification.
- Weight sharing is another major advantage of CNNs.
- Convolutional neural networks also minimize computation in comparisonwith a regular neural network.
- CNN‘s make use of the same knowledge across all image locations.

CHAPTER 5

SOFTWARE ENVIRONMENT

5.1 About Python

Below are some facts about Python.

1. Python is currently the most widely used multi-purpose, high-level programming language.
2. Python allows programming in Object-Oriented and Procedural paradigms. Python programs generally are smaller than other programming languages like Java.
3. Programmers have to type relatively less and indentation requirement of the language, makes them readable all the time.
4. Python language is being used by almost all tech-giant companies like – Google, Amazon, Facebook, Instagram, Dropbox, Uber… etc.

The biggest strength of Python is huge collection of standard libraries which can be used for the following –

- Machine Learning
- GUI Applications (like Kivy, Tkinter, PyQt etc.)
- Web frameworks like Django (used by YouTube, Instagram, Dropbox)
- Image processing (like Opencv, Pillow)
- Web scraping (like Scrapy, BeautifulSoup, Selenium)
- Test frameworks
- Multimedia

5.2 Advantages of Python

1. Extensive Libraries

Python downloads with an extensive library and it contain code for various purposes like regular expressions, documentation-generation, unit-testing, web browsers, threading, databases, CGI, email, image manipulation, and more. So, we don't have to write the complete code for that manually.

2. Extensible

As we have seen earlier, Python can be extended to other languages. You can write some of your code in languages like C++ or C. This comes in handy, especially in projects.

3. Embeddable

Complimentary to extensibility, Python is embeddable as well. You can put your Python code in your source code of a different language, like C++. This lets us add scripting capabilities to our code in the other language.

4. Improved Productivity

The language's simplicity and extensive libraries render programmers more productive than languages like Java and C++ do. Also, the fact that you need to write less and get more things done.

5. IOT Opportunities

Since Python forms the basis of new platforms like Raspberry Pi, it finds the future bright for the Internet of Things. This is a way to connect the language with the real world.

6. Simple and Easy

When working with Java, you may have to create a class to print 'Hello World'. But in Python, just a print statement will do. It is also quite easy to learn, understand, and code. This is why when people pick up Python, they have a hard time adjusting to other more verbose languages like Java.

7. Readable

Because it is not such a verbose language, reading Python is much like reading English. This is the reason why it is so easy to learn, understand, and code. It also does not need curly braces to define blocks, and indentation is mandatory. These further aids the readability of the code.

8. Object-Oriented

This language supports both the procedural and object-oriented programming paradigms. While functions help us with code reusability, classes and objects let us model the real world. A class allows the encapsulation of data and functions into one.

9. Free and Open-Source

Python is freely available. But not only can you download Python for free, but you can also download its source code, make changes to it, and even distribute it. It downloads with an extensive collection of libraries to help you with your tasks.

10. Portable

When code your project in a language like C++, you may need to make some changes to it if you want to run it on another platform. But it isn’t the same with Python. Here, you need to code only once, and you can run it anywhere. This is called Write Once Run Anywhere (WORA). However, you need to be careful enough not to include any system-dependent features.

11. Interpreted

Lastly it is said that it is an interpreted language. Since statements are executed one by one, debugging is easier than in compiled languages.

5.3 Advantages of Python Over Other Languages

1. Less Coding

Almost all of the tasks done in Python requires less coding when the same task is done in other languages. Python also has an awesome standard library support, so you don’t have to search for any third-party libraries to get your job done. This is the reason that many people suggest learning Python to beginners.

2. Affordable

Python is free therefore individuals, small companies or big organizations can leverage the free available resources to build applications. Python is popular and widely used so it gives you better community support.

The 2019 Github annual survey showed us that Python has overtaken Java in the most popular programming language category.

3. Python is for Everyone

Python code can run on any machine whether it is Linux, Mac or Windows. Programmers need to learn different languages for different jobs but with Python, you can professionally build web apps, perform data analysis and machine learning, automate things, do web scraping and also build games and powerful visualizations. It is an all-rounder programming language.

5.4 Required Packages

1.TensorFlow

TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library and is also used for machine learning applications such as neural networks. It is used for both research and production at Google.

TensorFlow was developed by the Google Brain team for internal Google use. It was released under the Apache 2.0 open-source license on November 9, 2015.

2.NumPy

NumPy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays.

It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

- A powerful N-dimensional array object
- Sophisticated (broadcasting) functions
- Tools for integrating C/C++ and Fortran code
- Useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary datatypes can be defined using NumPy which allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

3. Pandas

Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. Python was majorly used for data munging and preparation. It had very little contribution towards data analysis. Pandas solved this problem. Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data load, prepare, manipulate, model, and analyze. Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

4.Matplotlib

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shells, the Jupyter Notebook, web application servers, and four graphical user interface toolkits. Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, error charts, scatter plots, etc., with just a few lines of code. For examples, see the sample plots and thumbnail gallery.

For simple plotting the pyplot module provides a MATLAB-like interface, particularly when combined with IPython. For the power user, you have full control of line styles, font properties, axes properties, etc, via an object-oriented interface or via a set of functions familiar to MATLAB users.

5.Scikit – learn

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use. Python

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace.

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

- Python is Interpreted − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- Python is Interactive − you can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

Python also acknowledges that speed of development is important. Readable and terse code is part of this, and so is access to powerful constructs that avoid tedious repetition of code. Maintainability also ties into this may be an all but useless metric, but it does say something about how much code you have to scan, read and/or understand to troubleshoot problems or tweak behaviors. This speed of development, the ease with which a programmer of other languages can pick up basic Python skills and the huge standard library is key to another area where Python excels. All its tools have been quick to implement, saved a lot of time, and several of them have later been patched and updated by people with no Python background - without breaking.

5.5 Install Python Step-by-Step in Windows and Mac

Python a versatile programming language doesn't come pre-installed on your computer devices. Python was first released in the year 1991 and until today it is a very popular high-level programming language. Its style philosophy emphasizes code readability with its notable use of great whitespace.

The object-oriented approach and language construct provided by Python enables programmers to write both clear and logical code for projects. This software does not come pre-packaged with Windows.

How to Install Python on Windows and Mac?

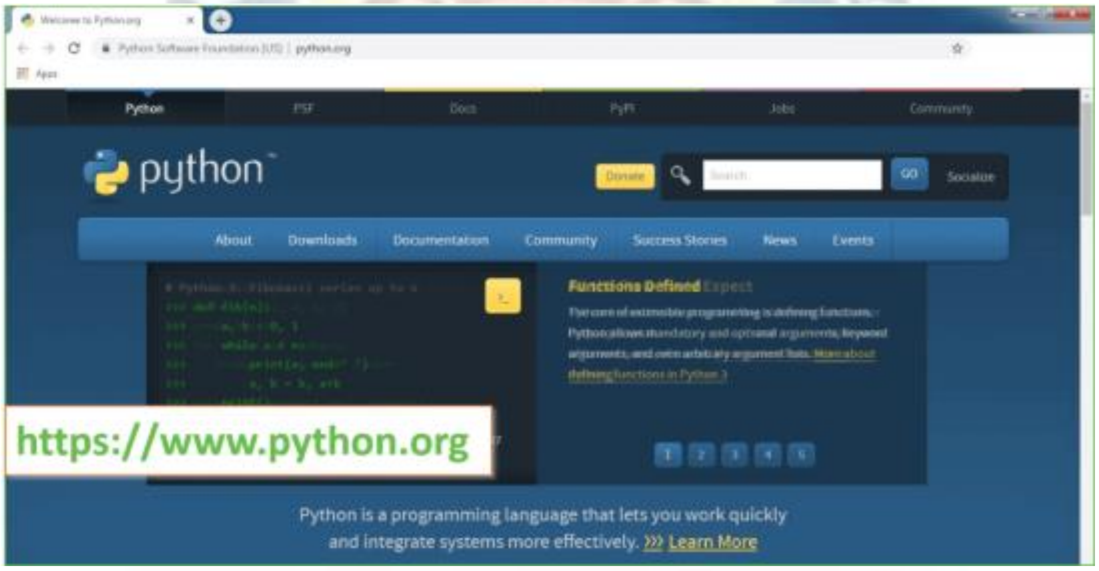
There have been several updates in the Python version over the years. The question is how to install Python? It might be confusing for the beginner who is willing to start learning Python but this tutorial will solve your query. The latest or the newest version of Python is version 3.7.4 or in other words, it is Python 3.

Note: The python version 3.7.4 cannot be used on Windows XP or earlier devices.

Before you start with the installation process of Python. First, you need to know about your System Requirements. Based on your system type i.e., operating system and based processor, you must download the python version. My system type is a Windows 64-bit operating system. So the steps below are to install python version 3.7.4 on Windows 7 device or to install Python 3. Download the Python Cheatsheet here. The steps on how to install Python on Windows 10, 8 and 7 are divided into 4 parts to help understand better.

Download the Correct version into the system

Step 1: Go to the official site to download and install python using Google Chrome or any other web browser. OR Click on the following link: https://www.python.org



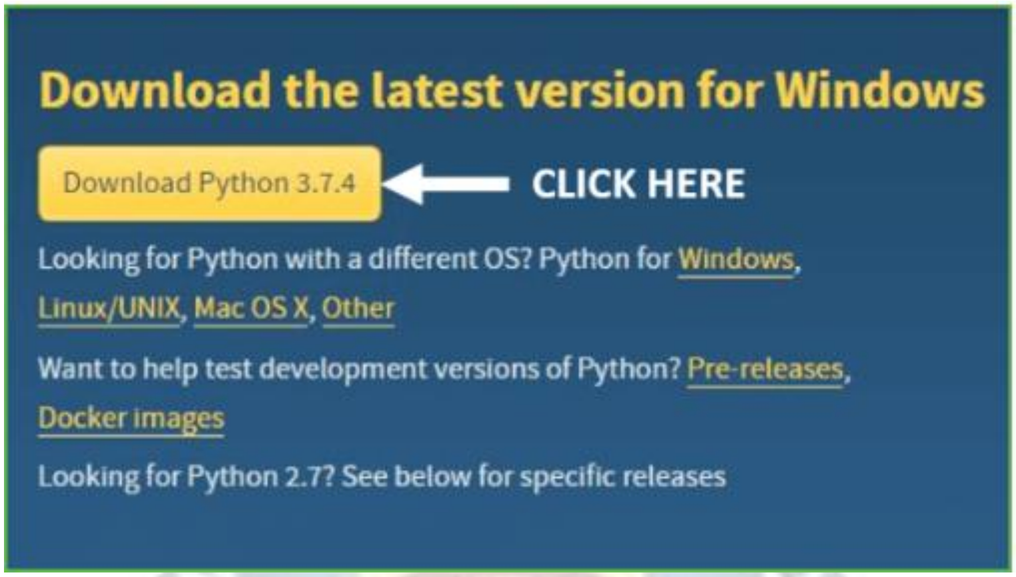
```
# Python 3: Fibonacci series up to n
>>> def fib(n):
>>>     a, b = 0, 1
>>>     while a < n:
>>>         print(a, end=' ')
>>>         a, b = b, a+b
```

The core of extensible programming is defining functions. Python allows mandatory and optional arguments, keyword arguments, and even arbitrary argument lists. More about defining functions in Python 3

Python is a programming language that lets you work quickly and integrate systems more effectively. >>> Learn More

Now, check for the latest and the correct version for your operating system.

Step 2: Click on the Download Tab.


Download the latest version for Windows

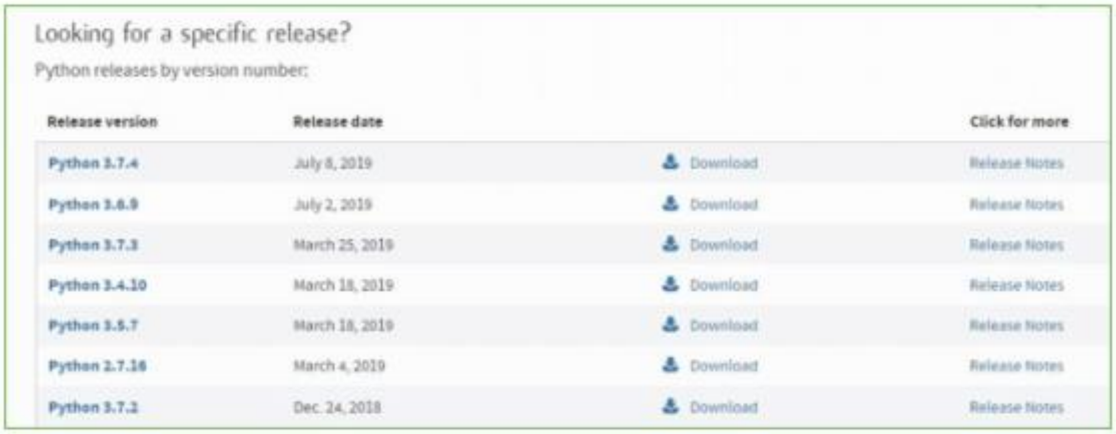
Download Python 3.7.4 ← CLICK HERE

Looking for Python with a different OS? Python for Windows, Linux/UNIX, Mac OS X, Other

Want to help test development versions of Python? Pre-releases, Docker images

Looking for Python 2.7? See below for specific releases

Step 3: You can either select the Download Python for windows 3.7.4 button in Yellow Color or you can scroll further down and click on download with respective to their version. Here, we are downloading the most recent python version for windows 3.7.4

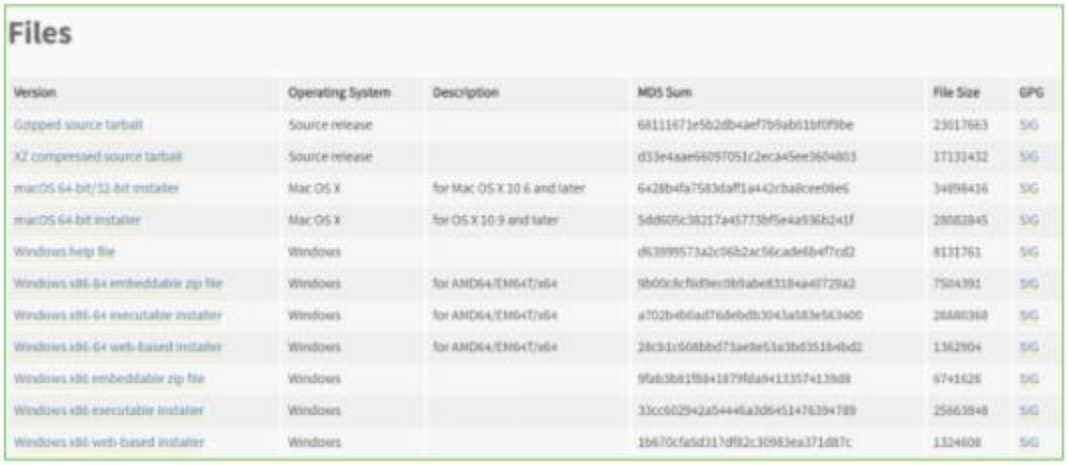

Looking for a specific release?

Python releases by version number:

Release version	Release date		Click for more
Python 3.7.4	July 8, 2019	Download	Release Notes
Python 3.6.9	July 2, 2019	Download	Release Notes
Python 3.7.3	March 25, 2019	Download	Release Notes
Python 3.4.10	March 18, 2019	Download	Release Notes
Python 3.5.7	March 18, 2019	Download	Release Notes
Python 2.7.16	March 4, 2019	Download	Release Notes
Python 3.7.2	Dec. 24, 2018	Download	Release Notes

Step 4: Scroll down the page until you find the Files option.

Step 5: Here you see a different version of python along with the operating system.



Files

Version	Operating System	Description	MD5 Sum	File Size	GPG
Gzipped source tarball	Source release		68111671e5b2db4aef7b9ab01bf0f9be	23017663	SIG
XZ compressed source tarball	Source release		d33e4aae66097051c2eca45ee3604803	17131432	SIG
macOS 64-bit/32-bit installer	Mac OS X	for Mac OS X 10.6 and later	6428b4fa7583daff1a442cba8cee08e6	34898416	SIG
macOS 64-bit installer	Mac OS X	for OS X 10.9 and later	5dd605c38217a45773bf5e4a936b241f	28082845	SIG
Windows help file	Windows		d63999573a2c06b2ac56cade6b4f7cd2	8131761	SIG
Windows x86-64 embeddable zip file	Windows	for AMD64/EM64T/x64	9b00c8cf6d9ec0b9abe83184a40729a2	7504391	SIG
Windows x86-64 executable installer	Windows	for AMD64/EM64T/x64	a702b4b0ad76debdb3043a583e563400	26680368	SIG
Windows x86-64 web-based installer	Windows	for AMD64/EM64T/x64	28cb1c608bbd73ae8e53a3bd351b4bd2	1362904	SIG
Windows x86 embeddable zip file	Windows		9fab3b81f8841879fda94133574139d8	6741626	SIG
Windows x86 executable installer	Windows		33cc602942a54446a3d6451476394789	25663848	SIG
Windows x86 web-based installer	Windows		1b670cfa5d317df82c30983ea371d87c	1324608	SIG

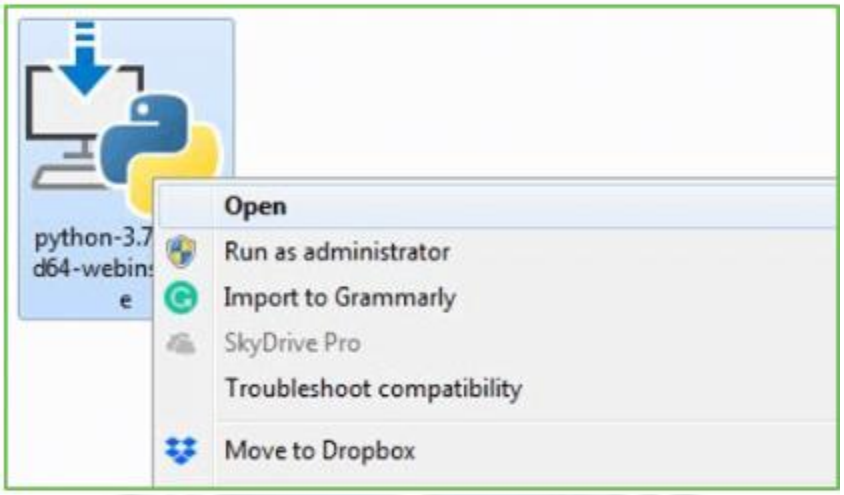
- To download Windows 32-bit python, you can select any one from the three options: Windows x86 embeddable zip file, Windows x86 executable installer or Windows x86 web-based installer.
- To download Windows 64-bit python, you can select any one from the three options: Windows x86-64 embeddable zip file, Windows x86-64 executable installer or Windows x86-64 web-based installer.

Here we will install Windows x86-64 web-based installer. Here your first part regarding which version of python is to be downloaded is completed. Now we move ahead with the second part in installing python i.e., Installation

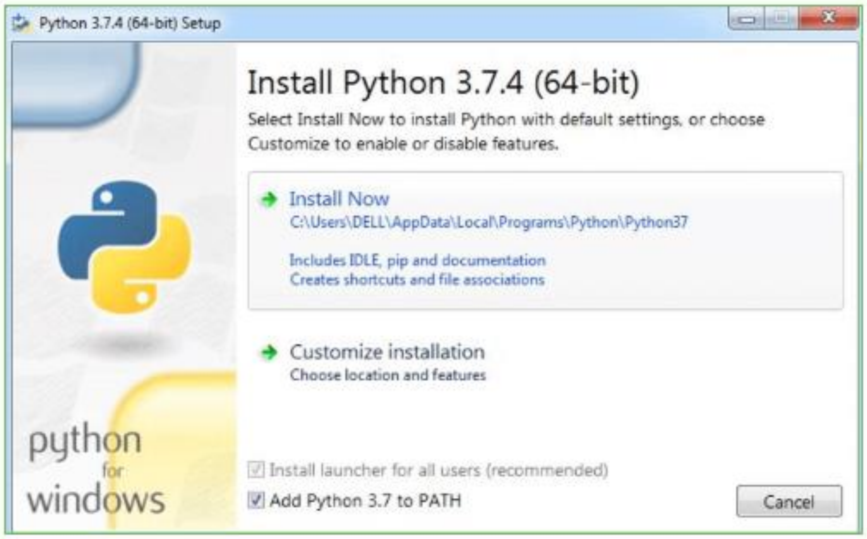
Note: To know the changes or updates that are made in the version you can click on the Release Note Option.

Installation of Python

Step 1: Go to Download and Open the downloaded python version to carry out the installation process.

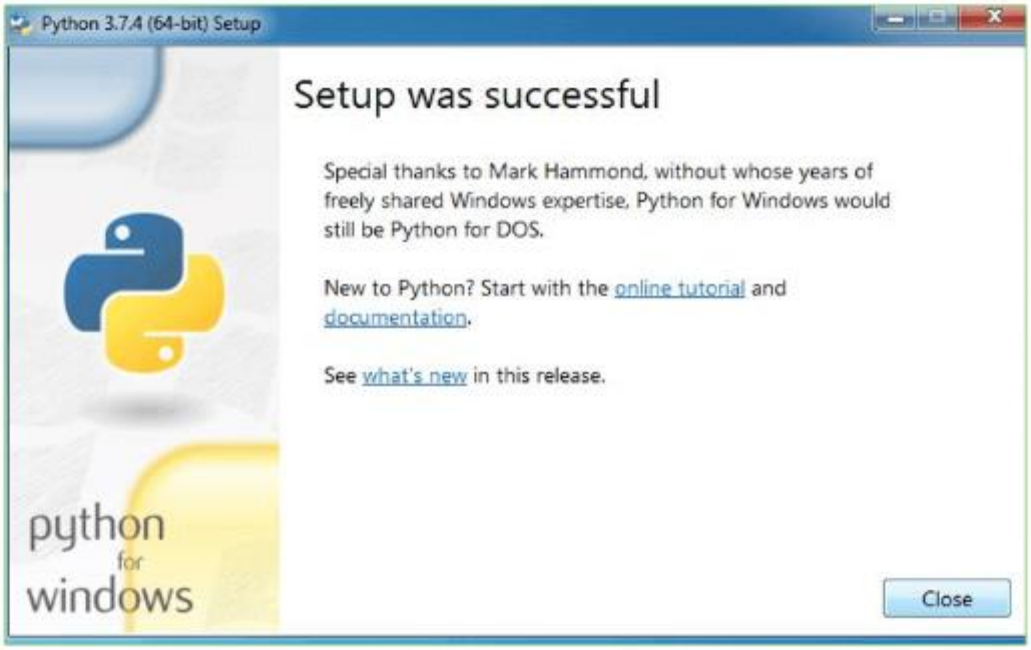


Step 2: Before you click on Install Now, Make sure to put a tick on Add Python 3.7 to PATH.


Python 3.7.4 (64-bit) Setup
Install Python 3.7.4 (64-bit)
Select Install Now to install Python with default settings, or choose Customize to enable or disable features.
Install Now
C:\Users\DELL\AppData\Local\Programs\Python\Python37
Includes IDLE, pip and documentation
Creates shortcuts and file associations
Customize installation
Choose location and features
python for windows
Cancel

[✓] Install launcher for all users (recommended)
[✓] Add Python 3.7 to PATH

Step 3: Click on Install NOW After the installation is successful. Click on Close.


Python 3.7.4 (64-bit) Setup

Setup was successful

Special thanks to Mark Hammond, without whose years of freely shared Windows expertise, Python for Windows would still be Python for DOS.

New to Python? Start with the online tutorial and documentation.

See what's new in this release.

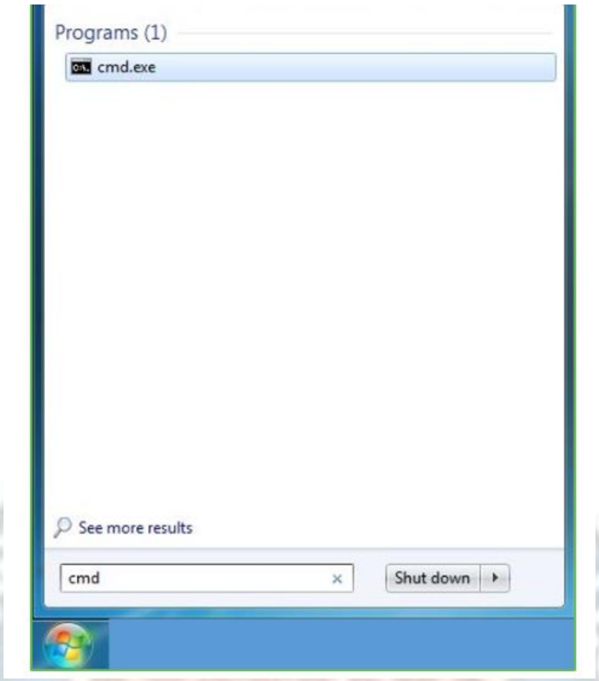
With these above three steps on python installation, you have successfully and correctly installed Python. Now is the time to verify the installation.

Note: The installation process might take a couple of minutes.

Verify the Python Installation

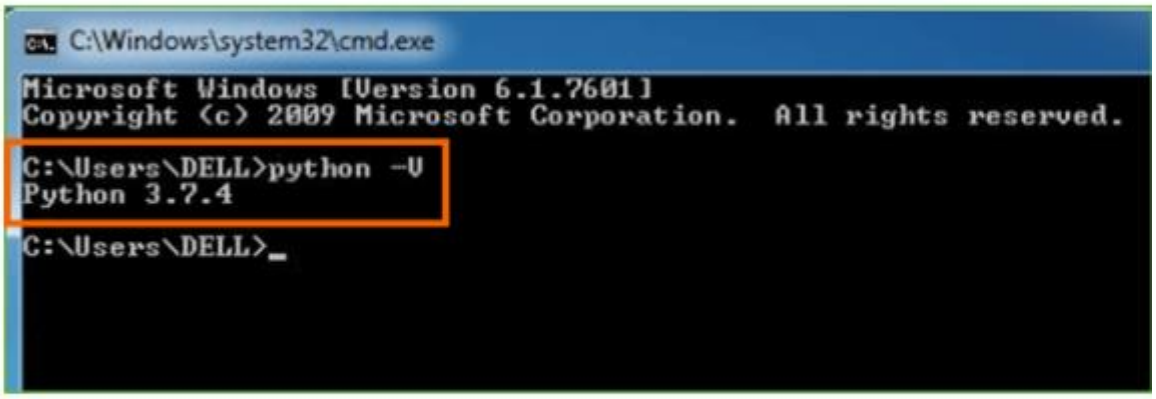
Step 1: Click on Start

Step 2: In the Windows Run Command, type “cmd”.



Step 3: Open the Command prompt option.

Step 4: Let us test whether the python is correctly installed. Type python –V and press Enter.



```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\DELL>python -V
Python 3.7.4

C:\Users\DELL>_
```

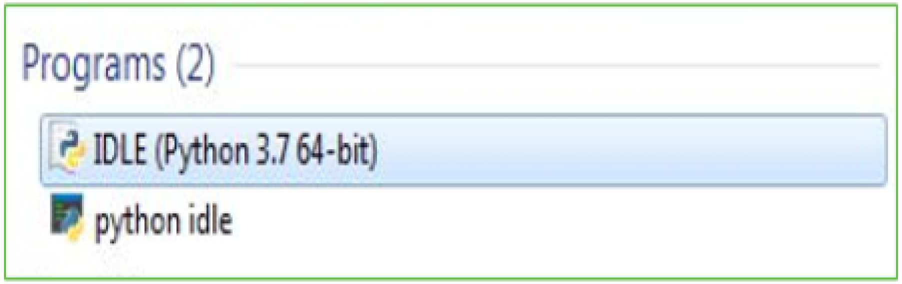
Step 5: You will get the answer as 3.7.4

Note: If you have any of the earlier versions of Python already installed. You must first uninstall the earlier version and then install the new one.

Check how the Python IDLE works

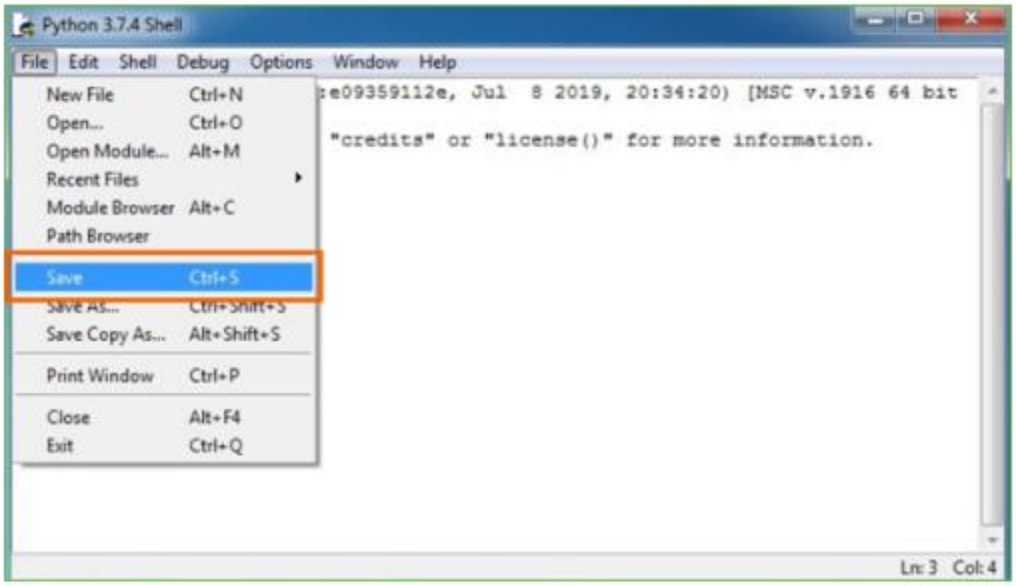
Step 1: Click on Start

Step 2: In the Windows Run command, type “python idle”.



Step 3: Click on IDLE (Python 3.7 64-bit) and launch the program

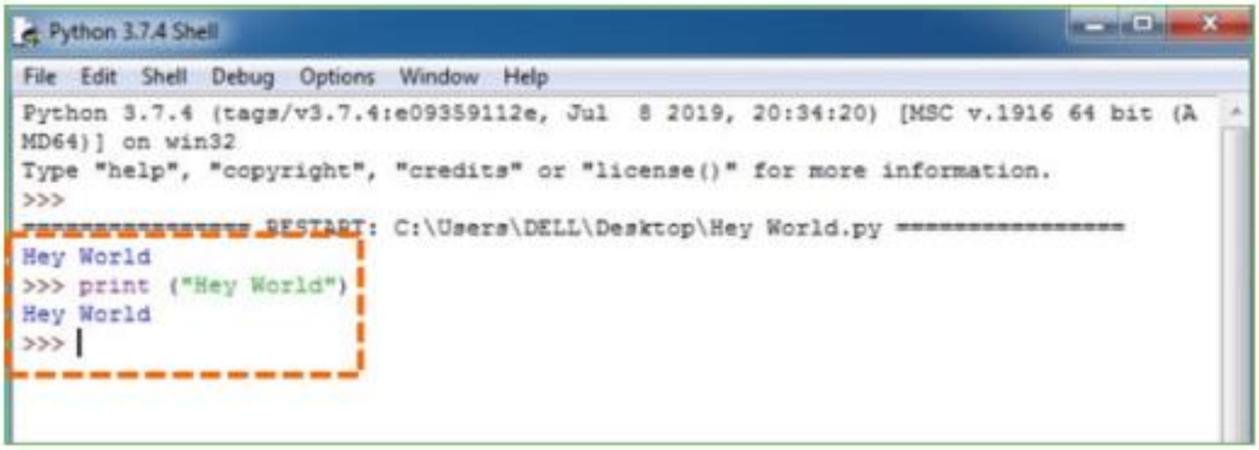
Step 4: To go ahead with working in IDLE you must first save the file. Click on File > Click on Save



```
e09359112e, Jul  8 2019, 20:34:20) [MSC v.1916 64 bit
"credits" or "license()" for more information.
```

Step 5: Name the file and save as type should be Python files. Click on SAVE. Here I have named the files as Hey World.

Step 6: Now for example enter print (“Hey World”) and Press Enter.



```
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul  8 2019, 20:34:20) [MSC v.1916 64 bit (A
MD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
================ RESTART: C:\Users\DELL\Desktop\Hey World.py ================
Hey World
>>> print ("Hey World")
Hey World
>>> 
```

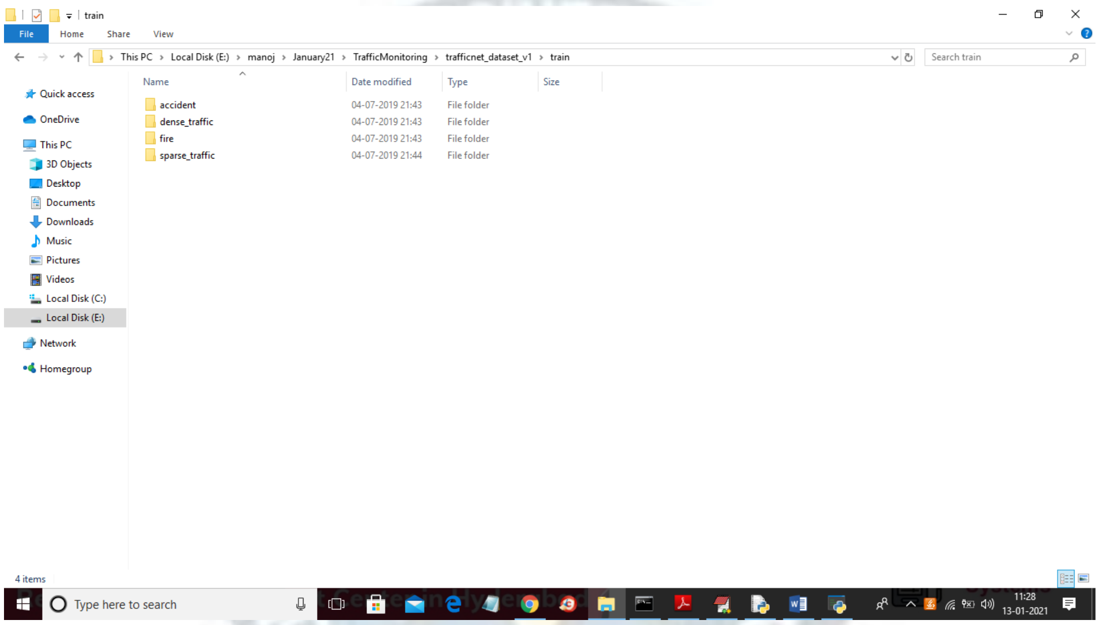
You will see that the command given is launched. With this, we end our tutorial on how to install Python. You have learned how to download python for windows into your respective operating system.

CHAPTER 6

RESULTS AND DISCUSSION

6.1 Results Obtained

This section gives the detailed analysis of simulation results implemented using “python environment”. Below is the dataset screen shots which contains various images

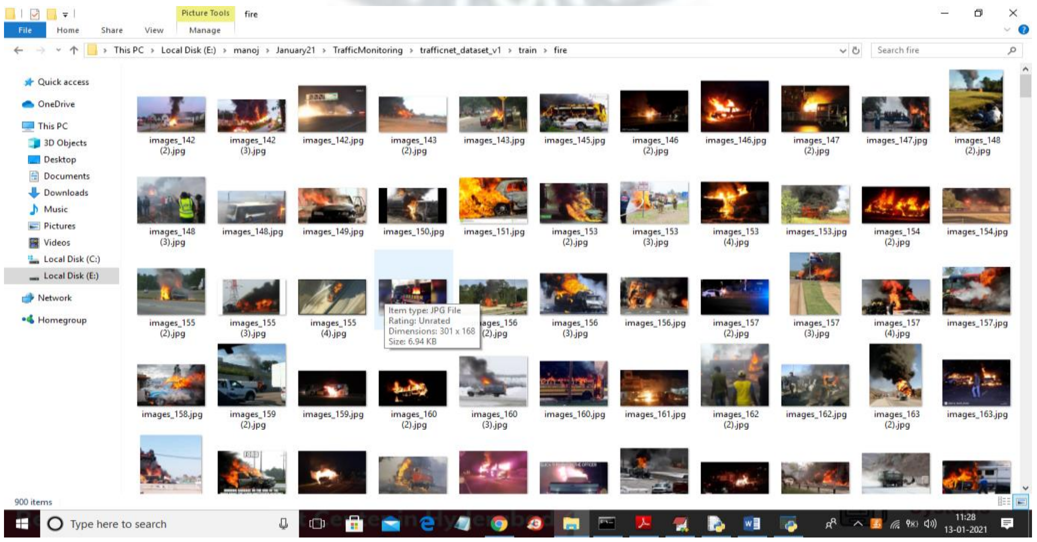


This PC › Local Disk (E:) › manoj › January21 › TrafficMonitoring › trafficnet_dataset_v1 › train

Name	Date modified	Type	Size
accident	04-07-2019 21:43	File folder	
dense_traffic	04-07-2019 21:43	File folder	
fire	04-07-2019 21:43	File folder	
sparse_traffic	04-07-2019 21:44	File folder	

4 items

- In above screen each folder contains its own images



This PC › Local Disk (E:) › manoj › January21 › TrafficMonitoring › trafficnet_dataset_v1 › train › fire

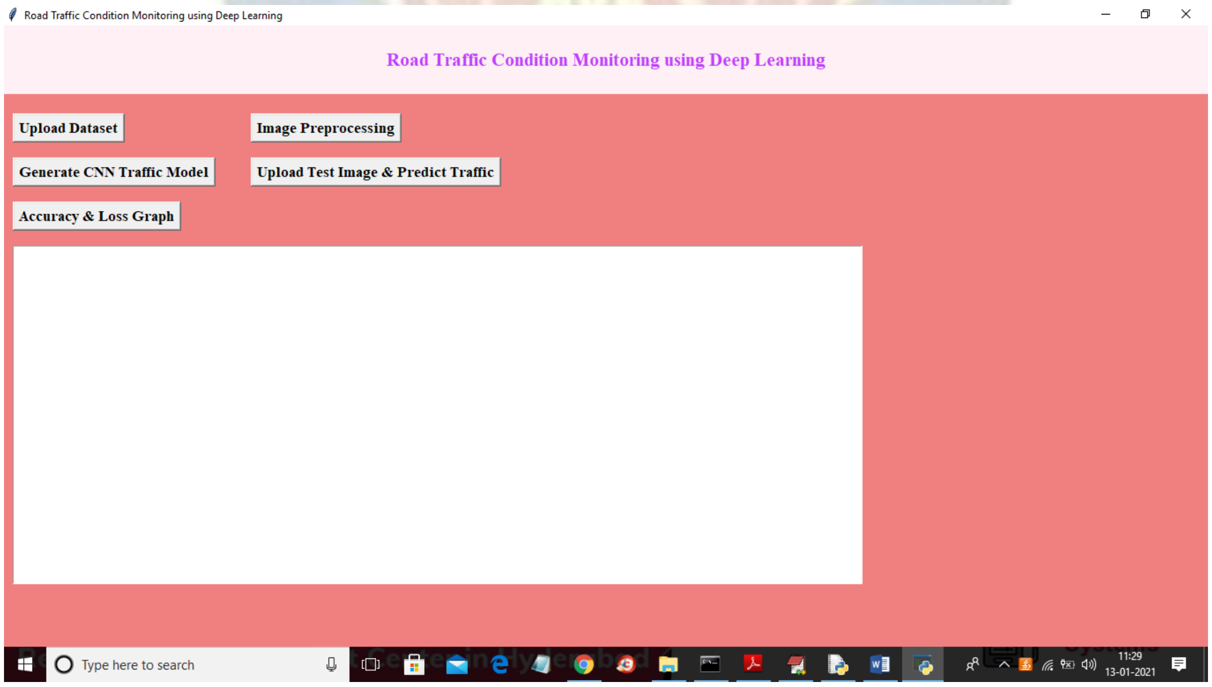
900 items

- The given screen showing some images from fire accident

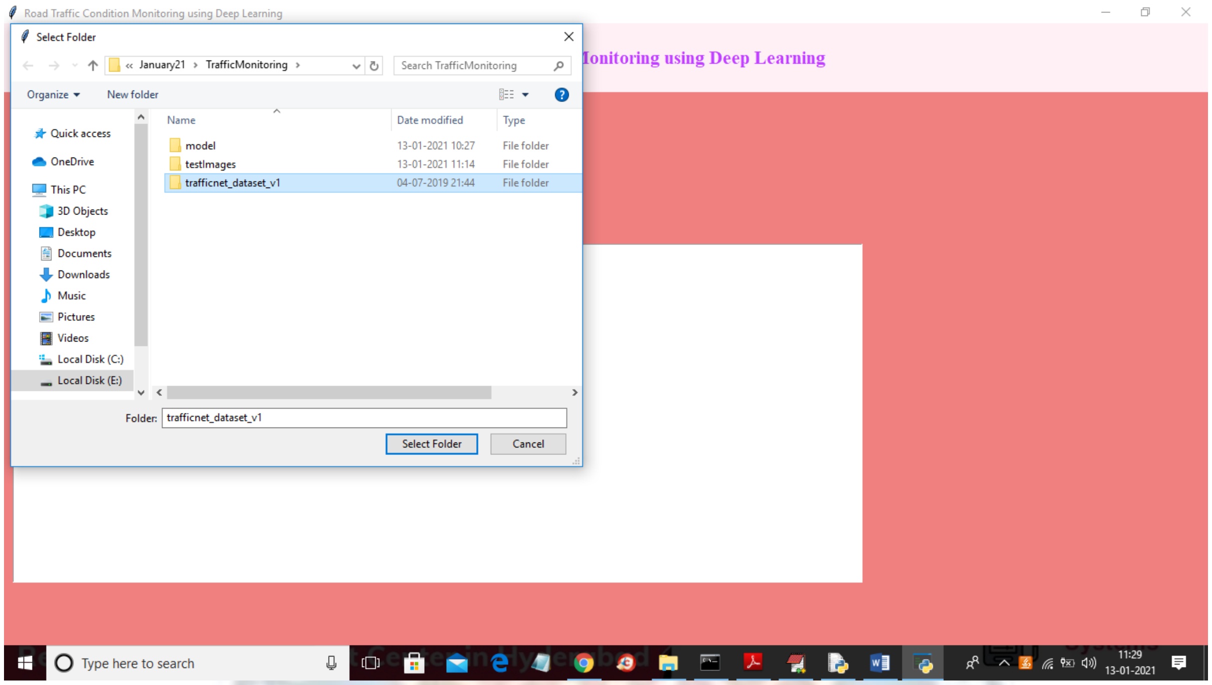
This project consists of following modules

1) Dataset Collection: To implement this project we have downloaded traffic dataset from GITHUB website. https://github.com/OlafenwaMoses/Traffic-Net/releases/tag/1.0
2) Preprocessing Image: Using this module we will read images from dataset and then resize all images and then extract pixels from images and convert all images to CNN compatible format
3) CNN Model Generation: using this module we will train CNN model with above processed images
4) Recognition Module: Using this module we will upload traffic image and then CNN model identify whether image contains heavy traffic, low traffic, fire accident and accident.

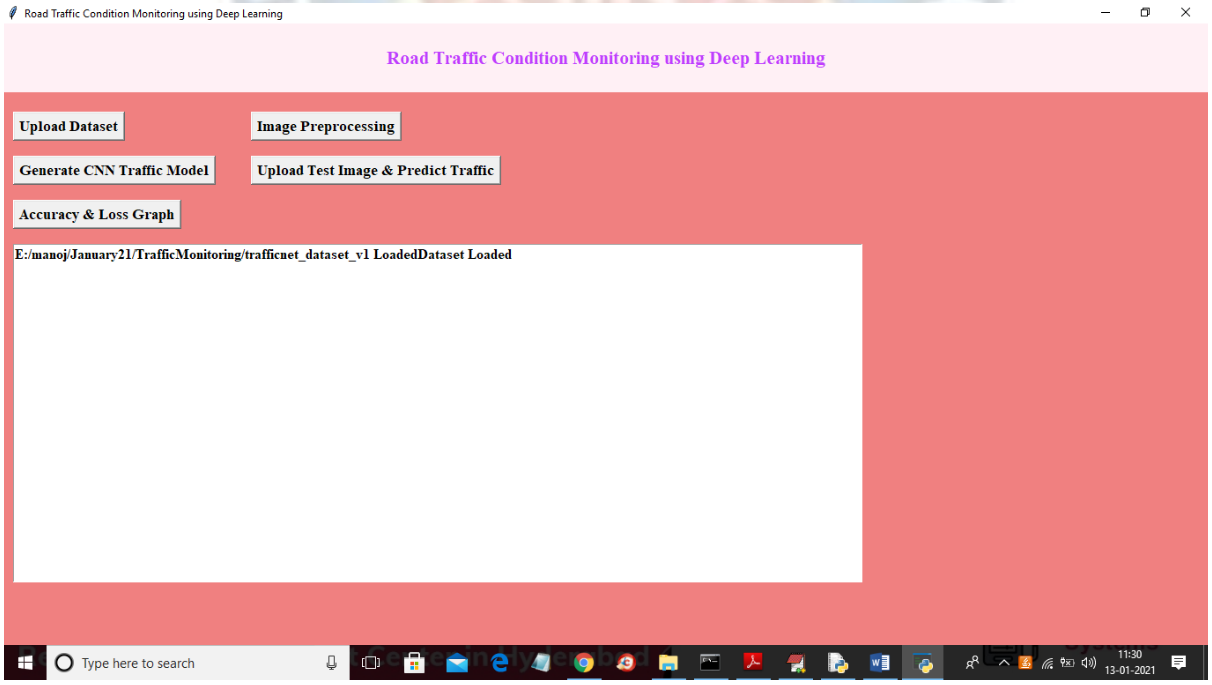
- Here are the screenshots of the result
- To run project double click on 'run.bat' file to get below screen


Road Traffic Condition Monitoring using Deep Learning
Road Traffic Condition Monitoring using Deep Learning
Upload Dataset
Image Preprocessing
Generate CNN Traffic Model
Upload Test Image & Predict Traffic
Accuracy & Loss Graph
Type here to search
11:29
13-01-2021

- In above screen click on 'Upload Dataset' button to load dataset

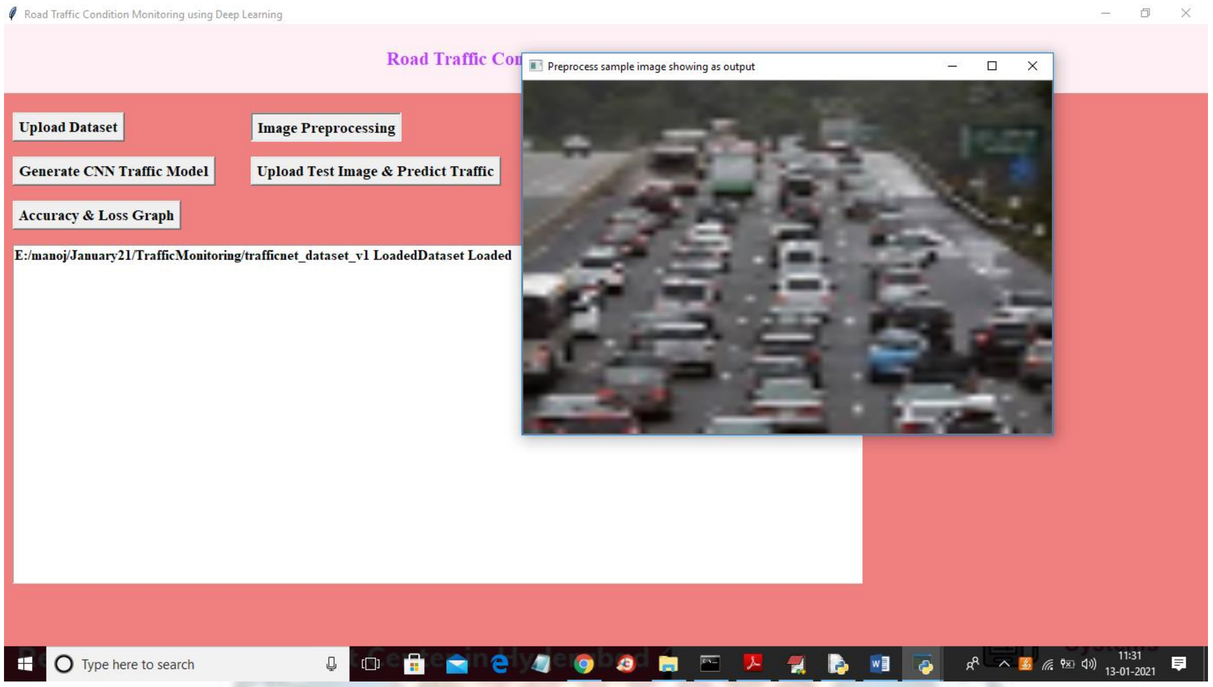


- In above screen selecting and uploading 'trafficnet_dataset' folder and then click on 'Select Folder' to load dataset and to get below screen


Road Traffic Condition Monitoring using Deep Learning
Road Traffic Condition Monitoring using Deep Learning
Upload Dataset
Image Preprocessing
Generate CNN Traffic Model
Upload Test Image & Predict Traffic
Accuracy & Loss Graph
Type here to search
11:30
13-01-2021

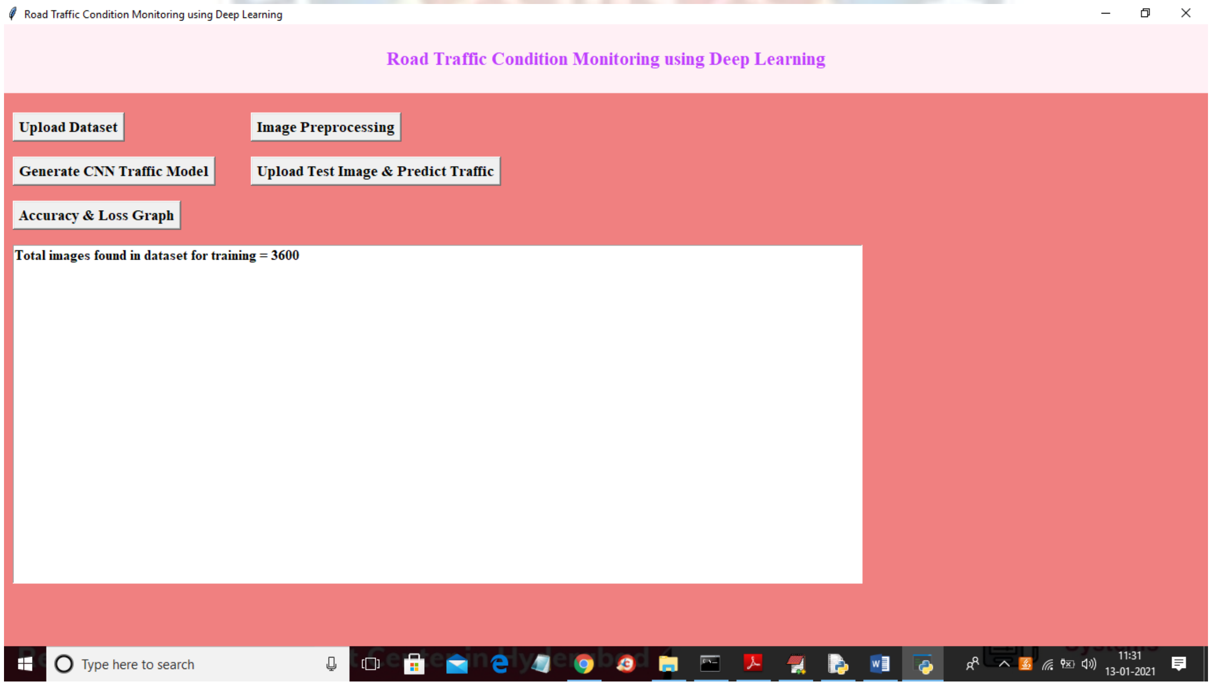
```
E:/manoj/January21/TrafficMonitoring/trafficnet_dataset_v1 LoadedDataset Loaded
```

- In above screen dataset loaded and then click on 'Image Preprocessing' button to read all images and then convert it CNN compatible format


Road Traffic Condition Monitoring using Deep Learning
Road Traffic Co
Preprocess sample image showing as output
Upload Dataset
Image Preprocessing
Generate CNN Traffic Model
Upload Test Image & Predict Traffic
Accuracy & Loss Graph
Type here to search
11:31
13-01-2021

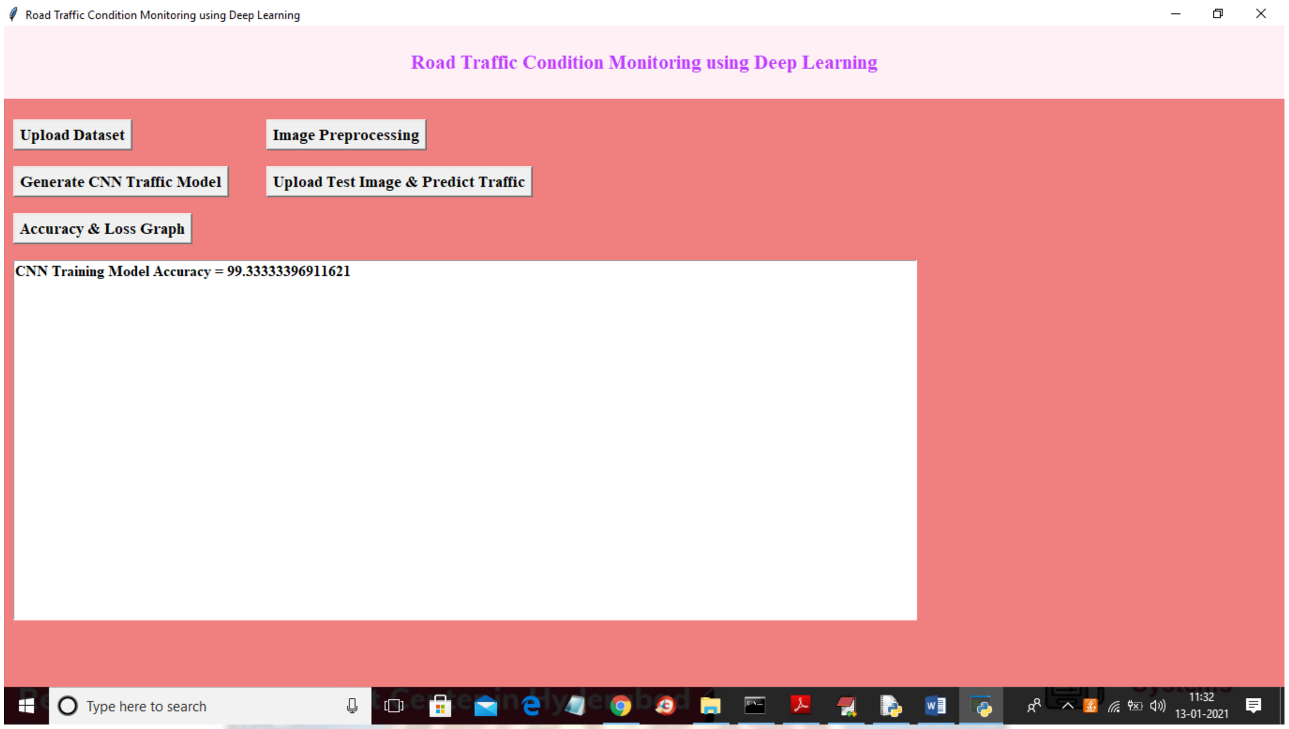
```
E:/manoj/January21/TrafficMonitoring/trafficnet_dataset_v1 LoadedDataset Loaded
```

- In above screen displaying sample image from processed images and now close above image to get below screen


Road Traffic Condition Monitoring using Deep Learning
Road Traffic Condition Monitoring using Deep Learning
Upload Dataset
Image Preprocessing
Generate CNN Traffic Model
Upload Test Image & Predict Traffic
Accuracy & Loss Graph
Type here to search
11:31
13-01-2021

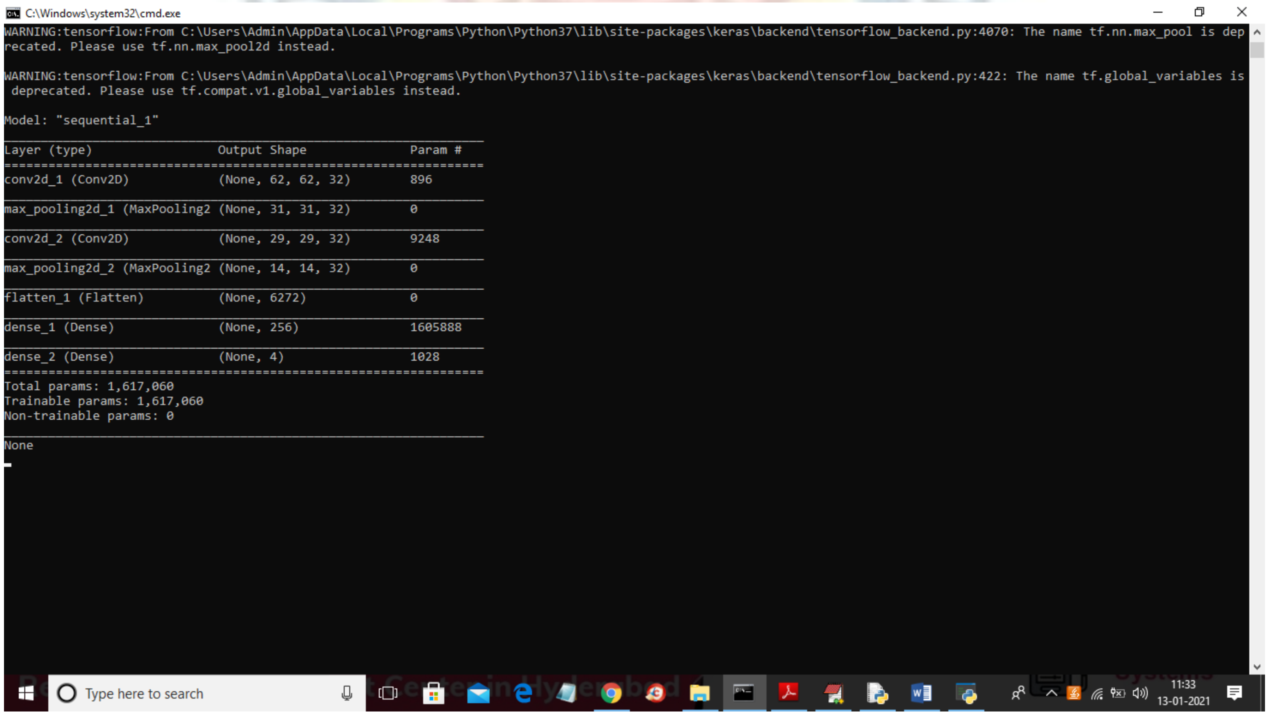
```
Total images found in dataset for training = 3600
```

- In above screen total processed images showing as 3600 and now click on ‘Generate CNN Traffic Model’ button to generate CNN model and to get below screen


Road Traffic Condition Monitoring using Deep Learning
Road Traffic Condition Monitoring using Deep Learning
Upload Dataset
Image Preprocessing
Generate CNN Traffic Model
Upload Test Image & Predict Traffic
Accuracy & Loss Graph
Type here to search
11:32
13-01-2021

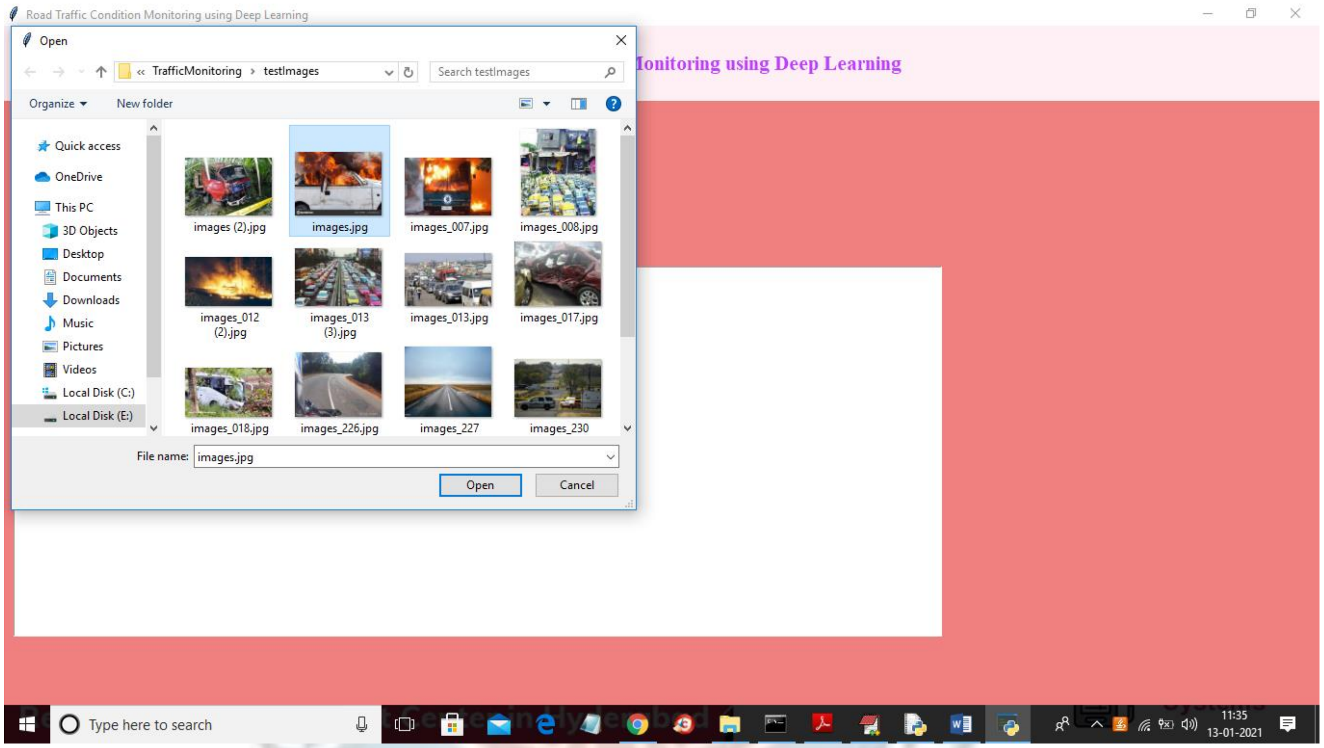
```
CNN Training Model Accuracy = 99.33333396911621
```

- In above screen CNN model generated with prediction accuracy as 99% and in below black console we can see CNN layer details



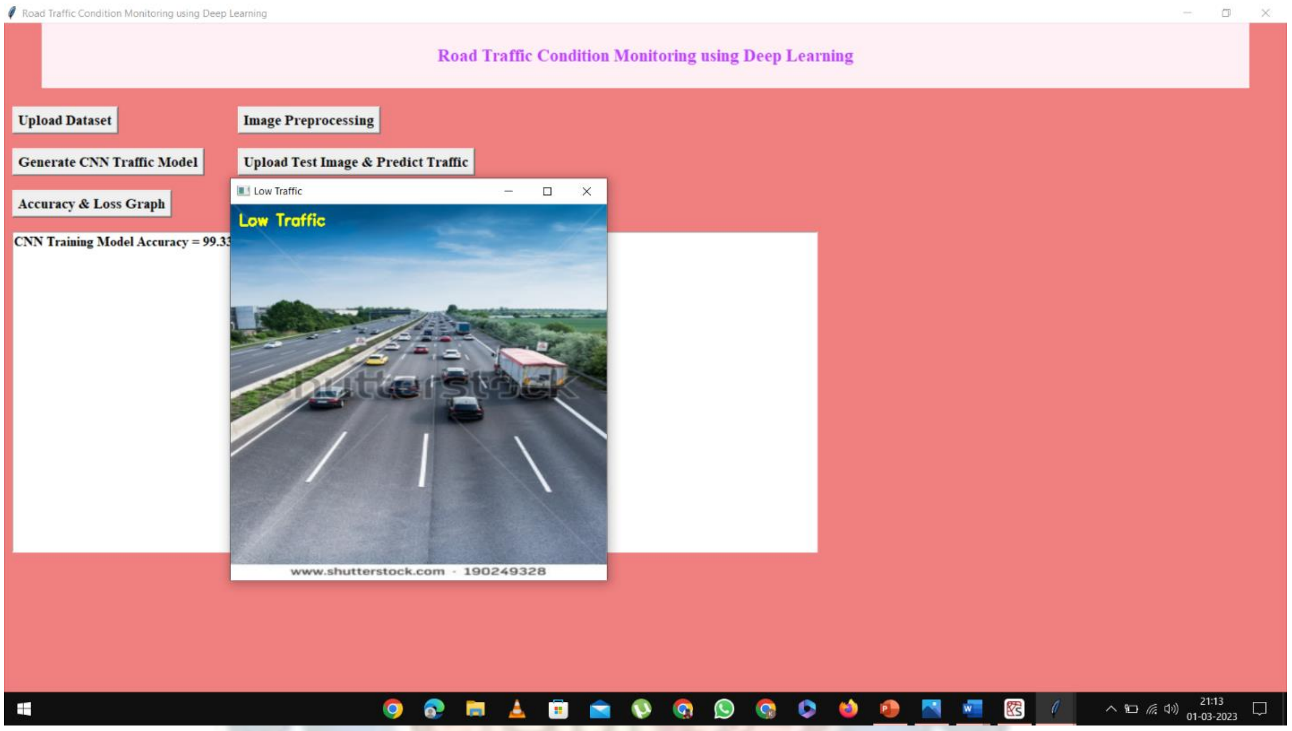
```
WARNING:tensorflow:From C:\Users\Admin\AppData\Local\Programs\Python\Python37\lib\site-packages\keras\backend\tensorflow_backend.py:4070: The name tf.nn.max_pool is dep
recated. Please use tf.nn.max_pool2d instead.

WARNING:tensorflow:From C:\Users\Admin\AppData\Local\Programs\Python\Python37\lib\site-packages\keras\backend\tensorflow_backend.py:422: The name tf.global_variables is
 deprecated. Please use tf.compat.v1.global_variables instead.

Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 62, 62, 32)        896
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 31, 31, 32)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 29, 29, 32)        9248
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 14, 14, 32)        0
_________________________________________________________________
flatten_1 (Flatten)          (None, 6272)              0
_________________________________________________________________
dense_1 (Dense)              (None, 256)               1605888
_________________________________________________________________
dense_2 (Dense)              (None, 4)                 1028
=================================================================
Total params: 1,617,060
Trainable params: 1,617,060
Non-trainable params: 0
_________________________________________________________________
None
```

- The given screen showing all layers of CNN model and each layer filtering images with different sizes such as 62 X 62, 31 X 31 etc. Now model is ready and now click on ‘Upload Test Image & Predict Traffic’ button and upload image to identify traffic condition



- In above screen selecting and uploading ‘images.jpg’ file and then click on ‘Open’ button to get below result.
- Here are the cases for the result

CASE 1: Low Traffic


Road Traffic Condition Monitoring using Deep Learning
Road Traffic Condition Monitoring using Deep Learning
Upload Dataset
Image Preprocessing
Generate CNN Traffic Model
Upload Test Image & Predict Traffic
Accuracy & Loss Graph
Low Traffic
Low Traffic
www.shutterstock.com · 190249328
21:13
01-03-2023

CNN Training Model Accuracy = 99.33

Fig 6.1 Classifier predicted as Low Traffic

- The above screen shows the image which indicates the low traffic.
- When we upload the image of Low Traffic the CNN classifier will perform the operation such as Image Processing, Generate CNN traffic model, upload test image and finally the classifier will predict the image as a Low Traffic.
- Also it Displays the accuracy and Loss graph of the corresponding image.

CASE 2: Heavy Traffic

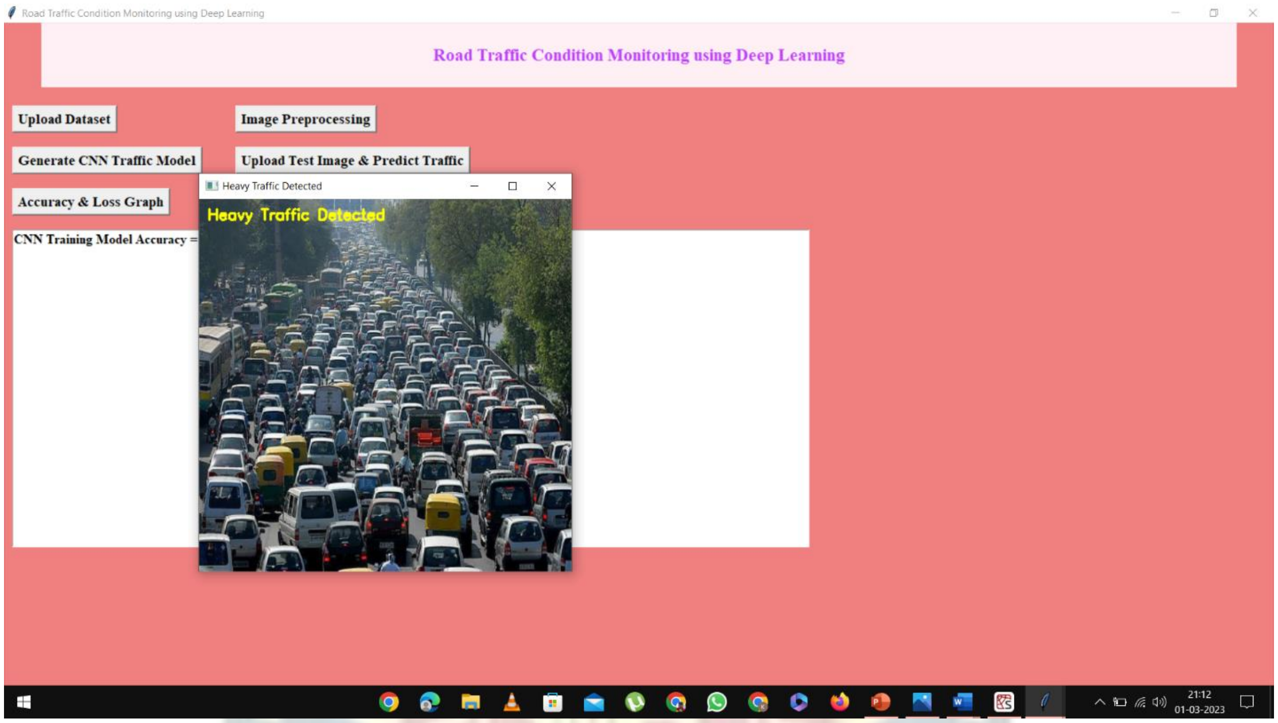
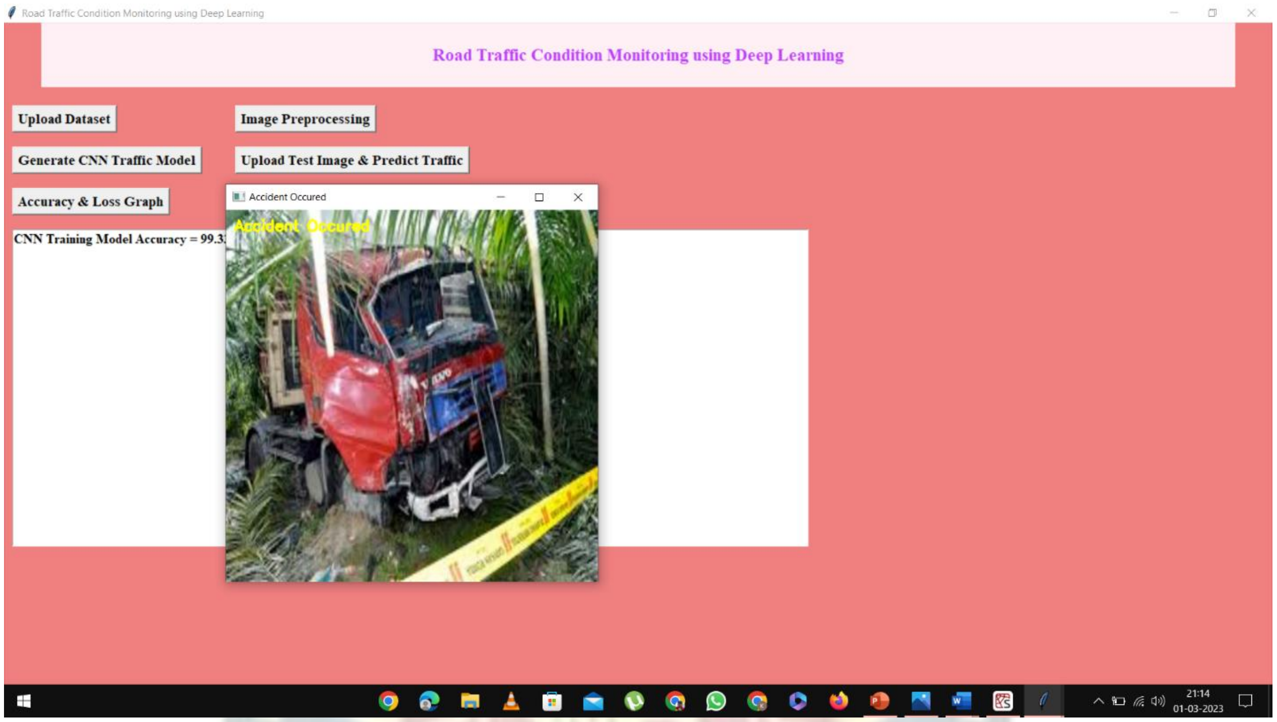

Road Traffic Condition Monitoring using Deep Learning
Road Traffic Condition Monitoring using Deep Learning
Upload Dataset
Image Preprocessing
Generate CNN Traffic Model
Upload Test Image & Predict Traffic
Accuracy & Loss Graph
CNN Training Model Accuracy =
Heavy Traffic Detected
Heavy Traffic Detected
21:12
01-03-2023

Fig 6.2 Classifier predicted as Heavy Traffic

- The above screen shows the image which indicates the High traffic.
- When we upload the image of High Traffic the CNN classifier will perform the operation such as Image Processing, Generate CNN traffic model, upload test image and finally the classifier will predict the image as Heavy Traffic.
- Also it Displays the accuracy and Loss graph of the corresponding image.

CASE 3: Accident Occurred


Road Traffic Condition Monitoring using Deep Learning
Road Traffic Condition Monitoring using Deep Learning
Upload Dataset
Image Preprocessing
Generate CNN Traffic Model
Upload Test Image & Predict Traffic
Accuracy & Loss Graph
Accident Occured
Accident Occured

CNN Training Model Accuracy = 99.3

Fig 6.3 Classifier predicted as Accident occurred

- The above screen shows the image which indicates the Accident occured.
- When we upload the image of Accident Occured the CNN classifier will perform the operation such as Image Processing, Generate CNN traffic model, upload test image and finally the classifier will predict the image as a Accident occured.
- Also it Displays the accuracy and Loss graph of the corresponding image.

CASE 4: Fire Accident Occurred

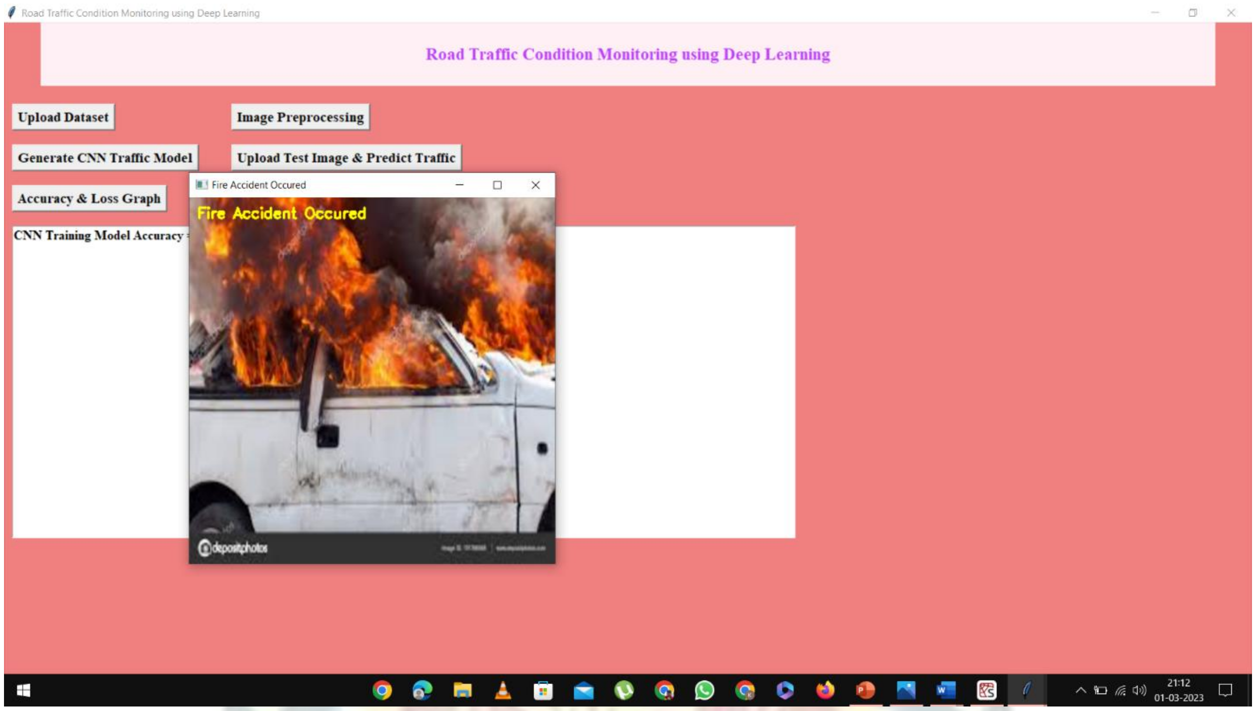

Road Traffic Condition Monitoring using Deep Learning
Road Traffic Condition Monitoring using Deep Learning
Upload Dataset
Image Preprocessing
Generate CNN Traffic Model
Upload Test Image & Predict Traffic
Accuracy & Loss Graph
CNN Training Model Accuracy
Fire Accident Occured
Fire Accident Occured
21:12
01-03-2023

Fig 6.4 Classifier predicted as Fire occurred

- The above screen shows the image which indicates the Fire Accident occured.
- When we upload the image of Fire Accident occured the CNN classifier will perform the operation such as Image Processing, Generate CNN traffic model, upload test image and finally the classifier will predict the image as a Fire Accident occured.
- Also it Displays the accuracy and Loss graph of the corresponding image.

6.2 Accuracy and Loss Graph

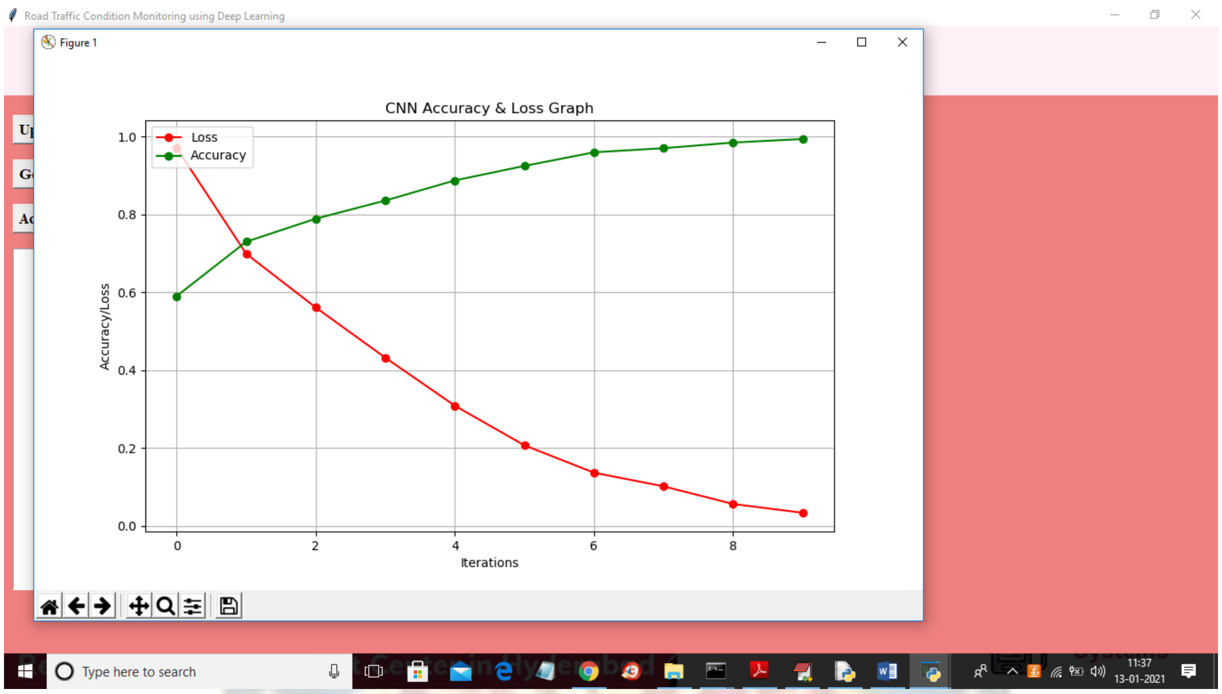


Fig 6.5 Accuracy and Loss Graph

- In above graph red line indicate LOSS and green line indicates accuracy and x-axis represents number of iterations or EPOCH and y-axis represents accuracy/loss. In above graph we can see in each iteration accuracy get increase and loss/error rate decrease which means in every iteration CNN model get better.

CHAPTER 7

CONCLUSION AND FUTURE SCOPE

7.1 Conclusion

The Convolution neural network is approached to identify the road condition without human intervention. The proposed CNN for training and validation is considered as a multi class problem i.e., low, heavy traffic , accident and fire accident occurred will be detected. Therefore we conclude that A MULTI PURPOSE ROAD SURVEILLANCE SYSTEM WITH DEEP LEARNING is a speedy system which is capable of handling the large scale traffic dataset with 99% efficiency.

7.2 Future Scope

As a future enhancement,

1. This is Purely software based project, if we combine this project with embedded system i.e, (Hardware) then this project will work efficiently and can send message to following Departments

- Hospitals – in case of accidents.
- Fire stations – in case of Fire accidents.
- This project can be efficiently work in managing the flow of traffic condition.

2. The traffic conditions are detected on the traffic videos on real-time. This can be done by video splitting technique are found and the traffic condition on every frame. Real time traffic detection on video is quite an important research for developing countries like India.

REFERENCES

1. Bashar, A. (2019). “Survey onEvolving Deep Learning Neural Network Architectures”. Journal of Artificial Intelligence, 1(02), 73-82.
2. Zoe Bartlett et al, “A Novel Online Dynamic Temporal Context Neural Network Framework for the Prediction of Road Traffic Flow” IEEE Access, vol.7, 2019.
3. Z. Zhao. Chen, X.Wu, P. C. Chen, and J. Liu, ``LSTM network: A deep learning approach for short-term traffic forecast," IET Intell. Transp. Syst., vol. 11, no. 2, pp. 6875, Mar. 2017.
4. T. Pamula, ``Road traffic conditions classification based on multilevel filtering of image content using convolutional neural networks," IEEE Intel. Transp. Syst. Mag., vol. 10, no. 3, pp. 1121, Jun. 2018.

APPENDIX

CODE

```
from tkinter import *
import tkinter
from tkinter import filedialog
import numpy as np
from tkinter.filedialog import askdirectory
from tkinter import simpledialog
import cv2
from keras.utils.np_utils import to_categorical
from keras.layers import Input
from keras.models import Model
from keras.layers import MaxPooling2D
from keras.layers import Dense, Dropout, Activation, Flatten
from keras.layers import Convolution2D
from keras.models import Sequential
import keras
import pickle
import matplotlib.pyplot as plt
import os
from keras.models import model_from_json

main = tkinter.Tk()
main.title("Road Traffic Condition Monitoring using Deep Learning") #designing main screen
main.geometry("1000x650")

global filename
global classifier
```

```
def upload():
  global filename
  filename = filedialog.askdirectory(initialdir = ".")
  text.delete('1.0', END)
  text.insert(END,filename+' Loaded')
  text.insert(END,"Dataset Loaded")

 def processImages():
    text.delete('1.0', END)
    X_train = np.load('model/X.txt.npy')
    Y_train = np.load('model/Y.txt.npy')
    text.insert(END,'Total images found in dataset for training = '+str(X_train.shape[0])+"\n\n")
    test = X_train[30]
    test = cv2.resize(test,(600,400))
    cv2.imshow('Preprocess sample image showing as output', test)
    cv2.waitKey(0)
    cv2.destroyAllWindows()

def generateModel():
  global classifier
  text.delete('1.0', END)
  if os.path.exists('model/model.json'):
      with open('model/model.json', "r") as json_file:
          loaded_model_json = json_file.read()
          classifier = model_from_json(loaded_model_json)
      classifier.load_weights("model/model_weights.h5")
      classifier._make_predict_function()
      print(classifier.summary())
      f = open('model/history.pckl', 'rb')
```

```
data = pickle.load(f)
      f.close()
      acc = data['accuracy']
      accuracy = acc[9] * 100
      text.insert(END,"CNN Training Model Accuracy = "+str(accuracy)+"\n")
  else:
      classifier = Sequential()
      classifier.add(Convolution2D(32, 3, 3, input_shape = (64, 64, 3), activation = 'relu'))
      classifier.add(MaxPooling2D(pool_size = (2, 2)))
      classifier.add(Convolution2D(32, 3, 3, activation = 'relu'))
      classifier.add(MaxPooling2D(pool_size = (2, 2)))
      classifier.add(Flatten())
      classifier.add(Dense(output_dim = 256, activation = 'relu'))
      classifier.add(Dense(output_dim = 4, activation = 'softmax'))
      print(classifier.summary())
      classifier.compile(optimizer = 'adam', loss = 'categorical_crossentropy', metrics = ['accuracy'])
      hist = classifier.fit(X_train, Y_train, batch_size=16, epochs=10, shuffle=True, verbose=2)
      classifier.save_weights('model/model_weights.h5')
      model_json = classifier.to_json()
      with open("model/model.json", "w") as json_file:
          json_file.write(model_json)
      f = open('model/history.pckl', 'wb')
      pickle.dump(hist.history, f)
      f.close()
      f = open('model/history.pckl', 'rb')
      data = pickle.load(f)
      f.close()
      acc = data['accuracy']
      accuracy = acc[9] * 100
text.insert(END,"CNN Training Model Accuracy = "+str(accuracy)+"\n")
```

```
def predictTraffic():
    name = filedialog.askopenfilename(initialdir="testImages")
    img = cv2.imread(name)
    img = cv2.resize(img, (64,64))
    im2arr = np.array(img)
    im2arr = im2arr.reshape(1,64,64,3)
    XX = np.asarray(im2arr)
    XX = XX.astype('float32')
    XX = XX/255
    preds = classifier.predict(XX)
    print(str(preds)+" "+str(np.argmax(preds)))
    predict = np.argmax(preds)
    print(predict)
    img = cv2.imread(name)
    img = cv2.resize(img,(450,450))
    msg = ''
    if predict == 0:
        cv2.putText(img, 'Accident Occured', (10, 25),  cv2.FONT_HERSHEY_SIMPLEX,0.6, (0, 255,
255), 2)
        msg = 'Accident Occured'
    if predict == 1:
        cv2.putText(img, 'Heavy Traffic Detected', (10, 25),  cv2.FONT_HERSHEY_SIMPLEX,0.6, (0,
255, 255), 2)
        msg = 'Heavy Traffic Detected'
    if predict == 2:
       cv2.putText(img, 'Fire Accident Occured', (10, 25),  cv2.FONT_HERSHEY_SIMPLEX,0.6, (0, 255,
255), 2)
        msg = 'Fire Accident Occured'
    if predict == 3:
        cv2.putText(img, 'Low Traffic', (10, 25),  cv2.FONT_HERSHEY_SIMPLEX,0.6, (0, 255, 255), 2)
        msg = 'Low Traffic'
    cv2.imshow(msg,img)
    cv2.waitKey(0)

def graph():
    f = open('model/history.pckl', 'rb')
    data = pickle.load(f)
    f.close()

    accuracy = data['accuracy']
    loss = data['loss']

    plt.figure(figsize=(10,6))
    plt.grid(True)
    plt.xlabel('Iterations')
    plt.ylabel('Accuracy/Loss')
    plt.plot(loss, 'ro-', color = 'red')
    plt.plot(accuracy, 'ro-', color = 'green')
    plt.legend(['Loss', 'Accuracy'], loc='upper left')
    plt.title('CNN Accuracy & Loss Graph')
    plt.show()

font = ('times', 16, 'bold')
title = Label(main, text='Road Traffic Condition Monitoring using Deep Learning', justify=LEFT)
title.config(bg='lavender blush', fg='DarkOrchid1')
title.config(font=font)
title.config(height=3, width=120)
title.place(x=100,y=5)
title.pack()

font1 = ('times', 13, 'bold')
```

```
uploadButton = Button(main, text="Upload Dataset", command=upload)
uploadButton.place(x=10,y=100)
uploadButton.config(font=font1)

processButton = Button(main, text="Image Preprocessing", command=processImages)
processButton.place(x=280,y=100)
processButton.config(font=font1)

cnnButton = Button(main, text="Generate CNN Traffic Model", command=generateModel)
cnnButton.place(x=10,y=150)
cnnButton.config(font=font1)

predictButton = Button(main, text="Upload Test Image & Predict Traffic", command=predictTraffic)
predictButton.place(x=280,y=150)
predictButton.config(font=font1)

graphButton = Button(main, text="Accuracy & Loss Graph", command=graph)
graphButton.place(x=10,y=200)
graphButton.config(font=font1)

font1 = ('times', 12, 'bold')
text=Text(main,height=20,width=120)
scroll=Scrollbar(text)
text.configure(yscrollcommand=scroll.set)
text.place(x=10,y=250)
text.config(font=font1)

main.config(bg='light coral')
main.mainloop()
```