ABSTRACT

Deep Learning has emerged as a new area in machine learning and is applied to a number of signal and image applications.The main purpose of the work presented in this paper, is to apply the concept of a Deep Learning algorithm namely, Convolutional neural networks (CNN) in image classification. The algorithm is tested on various standard datasets, like remote sensing data of aerial images (UC Merced Land Use Dataset) and scene images from SUN database. The performance of the algorithm is evaluated based on the quality metric known as Mean Squared Error (MSE) and classification accuracy. The graphical representation of the experimental results is given on the basis of MSE against the number of training epochs. The experimental result analysis based on the quality metrics and the graphical representation proves that the algorithm (CNN) gives fairly good classification accuracy for all datasets.

CHAPTER 1

INTRODUCTION

PROJECT DEFINITION

Deep learning object detection is a fast and effective way to predict an object's location in an image, Accident detection and traffic flow analysis are essential components of intelligent road traffic monitoring systems, contributing significantly to enhancing road safety, reducing congestion, and improving overall traffic management. These systems use a combination of advanced technologies and data analytics to achieve their objectives.

PROJECT OVERVIEW

This Project aims to enhance road flow using deep learning by collecting and preprocessing a diverse dataset, designing arobust CNN architecture, and integrating real-time data sources, the system will accurately identify road traffic condition. Data argumentation techniques and thorough testing willensure the model's reliability. The outcome is a deployable solution contributing to efficient road maintenance and improved safety.

SOFTWARE REQUIREMENTS:

- Python
- Django
- Mysql
- Wampserver

HARDWARE REQUIREMENTS

- Processor: Pentium IV or higher
- RAM: 256 MB
- Space on Hard Disk: minimum 512MB

CHAPTER 2

LITERATURE SURVEY

Existing System:

Convolutional neural systems (CNN) have been generally utilized in programmed picture classification frameworks. As a rule, highlights from the top layer of the CNN are used for classification; be that as it may, those highlights may not contain enough valuable data to foresee a picture effectively. Now and again, highlights from the lower layer convey more discriminative force than those from the top. Along these lines, applying highlights from a specific layer just to classification is by all accounts a procedure that doesn't use took in CNN's potential discriminant capacity to its full degree. This intrinsic property prompts the requirement for combination of highlights from various layers.

Proposed System:

We propose a strategy for consolidating highlights from different layers in given CNN models. In addition, effectively learned CNN models with preparing pictures are reused to separate highlights from numerous layers. The proposed combination strategy is assessed by picture classification benchmark informational indexes, CIFAR-10, NORB, and SVHN. In all cases, we show that the proposed strategy improves the detailed exhibitions of the current models by 0.38%, 3.22% and 0.13%, separately.

Advantages:

1) Spatial Hierarchy of features
2) Paramaters sharing
3) Translation Invariance

Disadvantages:

1) High cost of set up and equipment.
2) Continuous maintenance.

3) Complex algorithms involved.

CHAPTER 3

METHODOLOGY

Deep Learning:

Deep learning is a branch of machine learning which is completely based on artificial neural networks, as neural network is going to mimic the human brain so deep learning is also a kind of mimic of human brain. In deep learning, we don't need to explicitly program everything. The concept of deep learning is not new. It has been around for a couple of years now. It's on hype nowadays because earlier we did not have that much processing power and a lot of data. As in the last 20 years, the processing power increases exponentially, deep learning and machine learning came in the picture. A formal definition of deep learning is- neurons.

Deep Learning is a subset of Machine Learning that is based on artificial neural networks(ANNs) with multiple layers ,also known as deep neural networks (DNNs).ThemethodologyforimplementingaConvolutionalNeuralNetwork(CNN) algorithm using deep learning involves several sequential steps. It begins with the collection of a well-labeled dataset, followed by preprocessing tasks such as data cleaning, normalization, and resizing. The dataset is then divided into training and testing sets to assess the model's generalization ability.

The design of the CNN architecture is a critical step, incorporating convolutional layers for feature extraction and pooling layers for spatial reduction. Model compilation involves selecting appropriate loss functions and optimization algorithms, followed by training the model on the training dataset while carefully monitoring its performance on a validation set. Techniques like drop out and batch normalization are often applied to prevent overfitting.

To enhance the model's robustness, data augmentation techniques are employed to artificially increase the diversity of the training set. After training, the model's performance is evaluated on the test set using metrics such as accuracy, precision, recall, and F1 score. Misclassifications are analyzed for potential improvements, leading to fine-tuning if necessary.
Once the model achieves satisfactory performance, it can be deployed for real-world applications. Considerations for deployment include factors like latency, scalability, and resource efficiency. Continuous monitoring ensure the model's effectiveness.

Overtime ,and updates are implemented as needed to adapt to changing data patterns or enhance over all accuracy. This comprehensive methodology ensures a systematic and effective approach to building and deploying CNN algorithms for deep learning tasks.

These neural networks are inspired by the structure and function of the human brain, and they are designed to learn from large amounts of data in an unsupervised or semi-supervised manner.

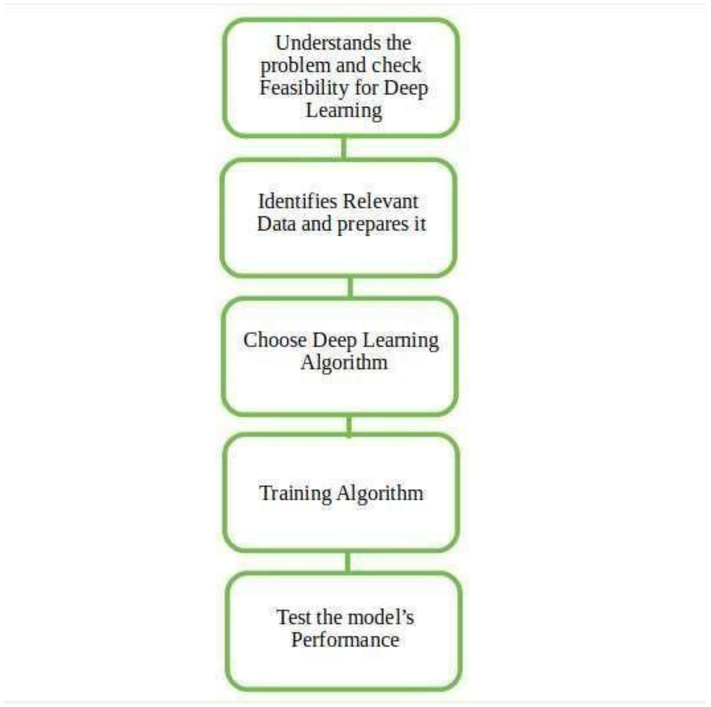
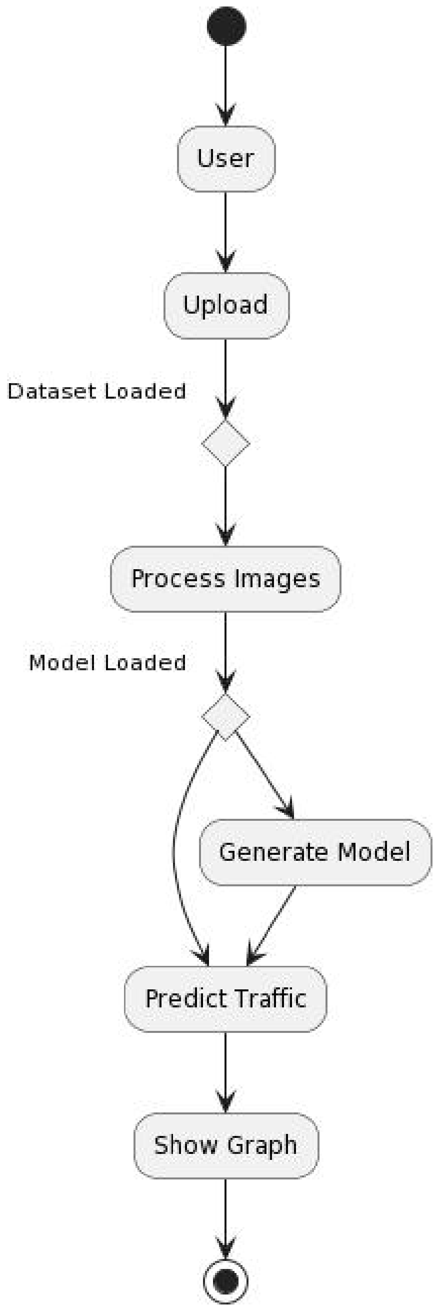

Understands the problem and check Feasibility for Deep Learning
Identifies Relevant Data and prepares it
Choose Deep Learning Algorithm
Training Algorithm
Test the model's Performance

Fig-Flow chart to understand Deep Learning

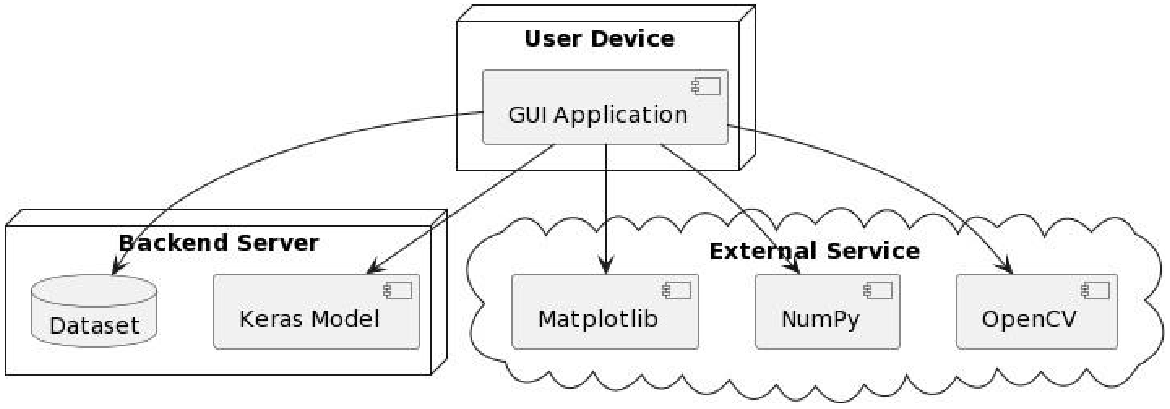
1.3. PROCESS MODEL



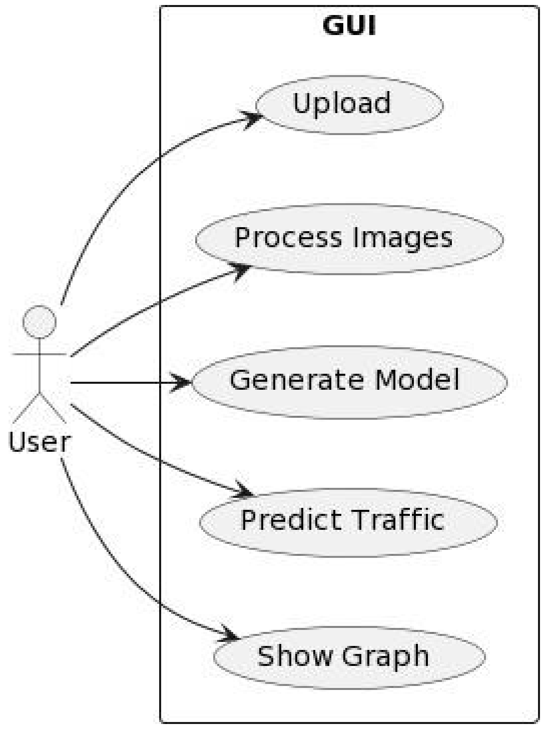
CHAPTER 4

SYSTEM ARCHITECTURE

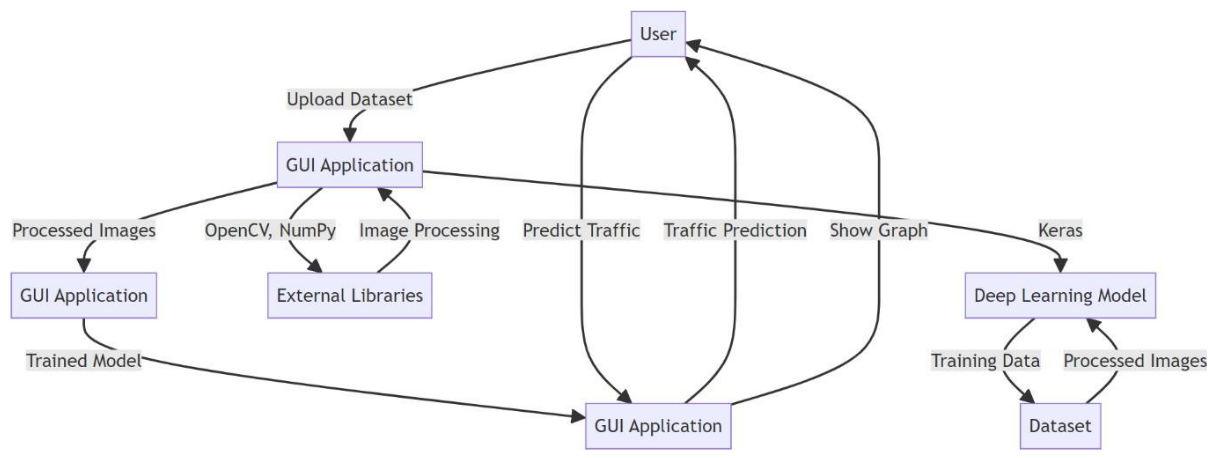
Deployment diagram



Use case diagram

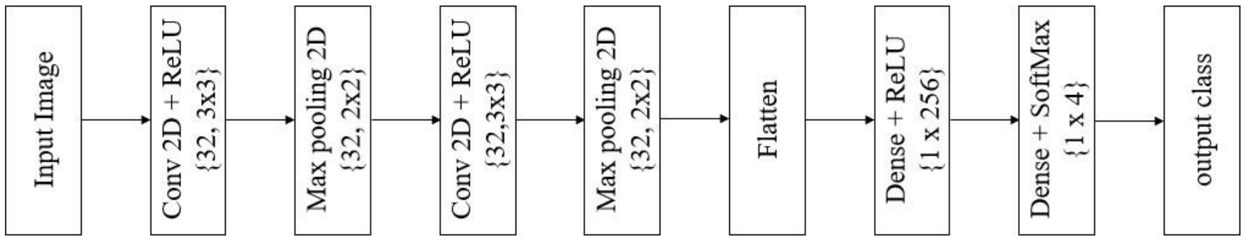


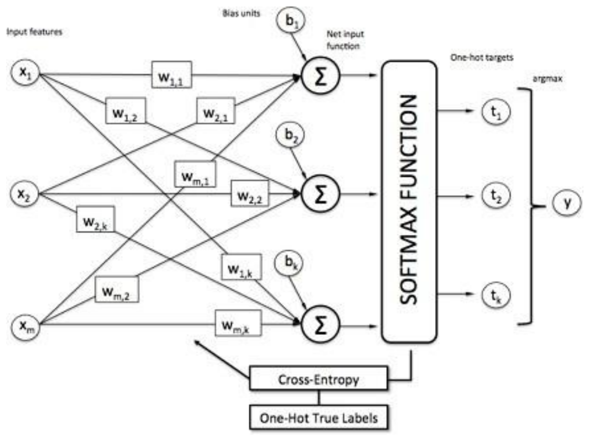
Designing Stage – Dataflow Diagram

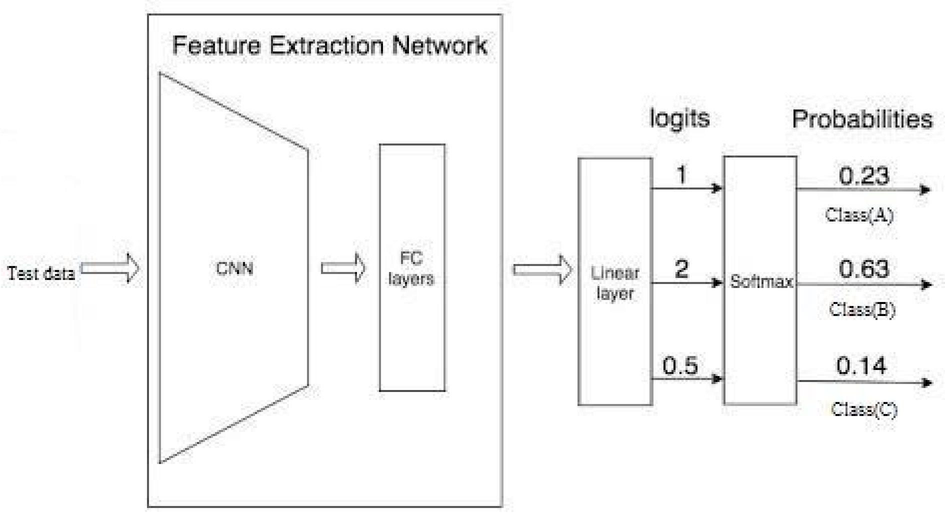


CHAPTER 5
IMPLEMENTATION

Implementation and Testing:







SOURCE CODE

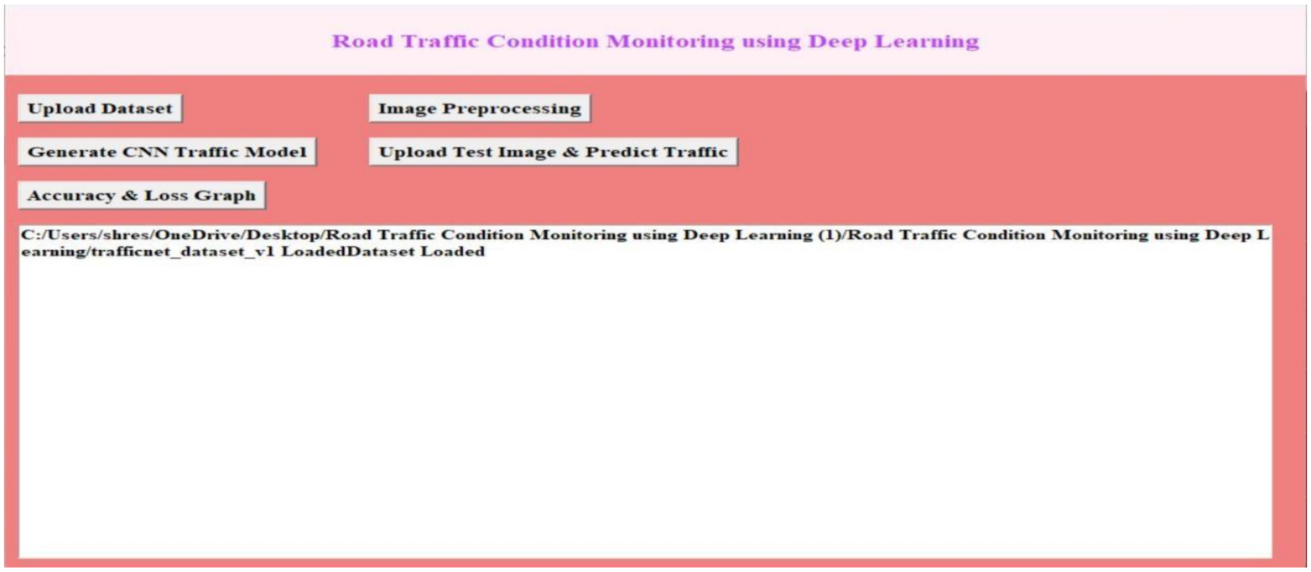
```
from tkinter import *
import tkinter
from tkinter import filedialog
import numpy as np
from tkinter.filedialog import askdirectory
from tkinter import simpledialog
import cv2
from keras.utils.np_utils import to_categorical
from keras.layers import Input
from keras.models import Model
from keras.layers 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()
```

CHAPTER 6

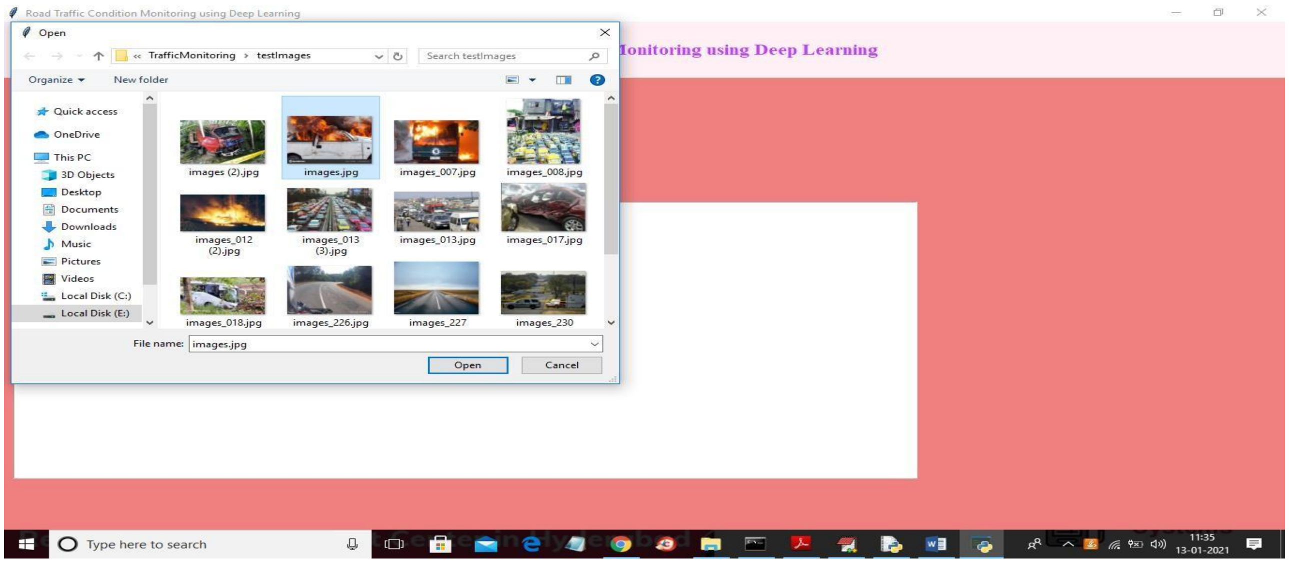
RESULTS

To run project double clickon 'run .bat 'file to get below screen


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

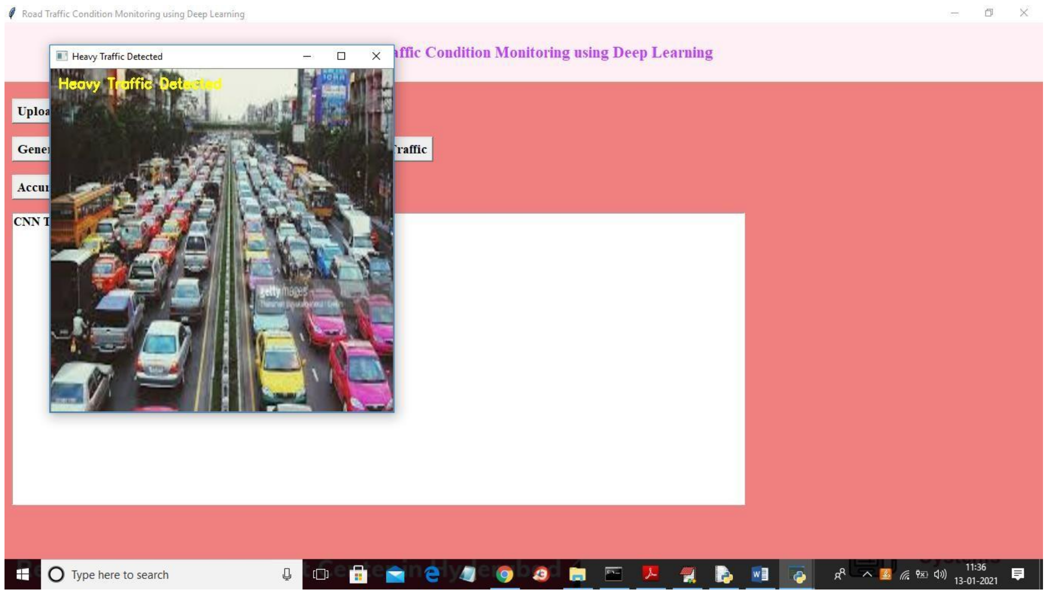
```
C:/Users/shres/OneDrive/Desktop/Road Traffic Condition Monitoring using Deep Learning (1)/Road Traffic Condition Monitoring using Deep L
earning/trafficnet_dataset_v1 LoadedDataset Loaded
```

In above screen CNN model generated and now click on 'Upload Test Image & predict traffic' button to upload test image and to get below result

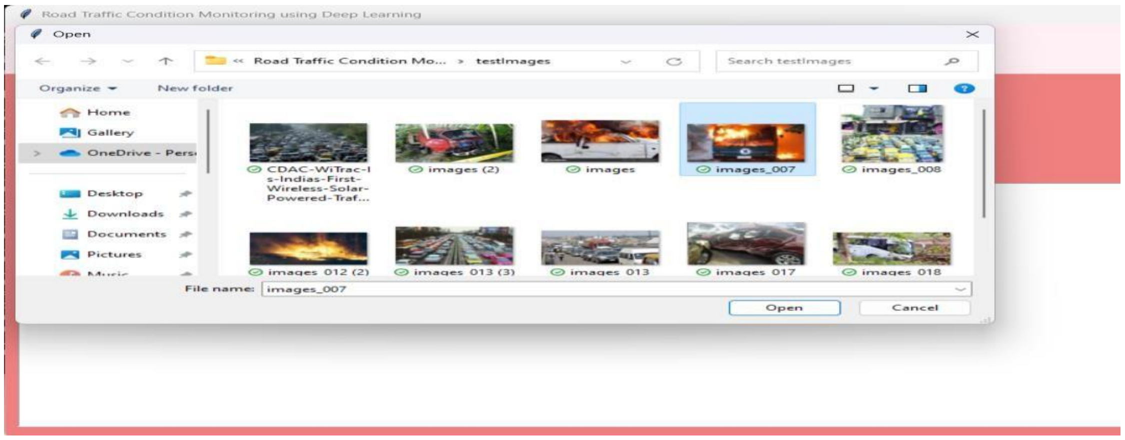


Identify objects:

In above screen selecting and uploading '1.jpg' file and then click on 'Open' button to get below screen



In above screen uploaded image classified as 'heavy traffic detected' and now test with another image



CHAPTER 7

CONCLUSION AND FUTURE SCOPE

In conclusion, the proposed method for traffic status prediction using DLCNN on the Traffic Net dataset presents a promising approach to addressing the challenges of traffic management, safety, and urban planning. This method harnesses the power of deep learning to accurately classify traffic conditions, offering advantages such as automation, scalability, and real-time monitoring. By providing precise insights into traffic dynamics, it enhances road safety, reduces congestion, and contributes to more efficient transportation systems. The model's adaptability and potential for continuous improvement make it a valuable tool for addressing the ever-evolving challenges of urban traffic. As cities continue to grow and traffic complexity increases, the significance of such methods cannot be overstated, offering a path towards smarter and safer urban mobility.

FUTURE SCOPE

- **Real-Time Traffic Management Systems**: Building upon this research, future applications could involve the development of real-time traffic management systems that not only predict traffic conditions but also dynamically control traffic signals, route traffic, and coordinate emergency responses based on predictive models.
- **Privacy and Security**: As traffic monitoring systems become more sophisticated, ensuring the privacy and security of collected data will be paramount. Research in this area should focus on robust encryption, data anonymization techniques, and secure communication protocols.There are many uses by this project like
- Advanced Deep Learning Architectures
- Multimodal Data Fusion
- Edge Computing and IoT Integration
- Urban Mobility Solutions

REFERENCES

[1] Lillesand, T.M. and Kiefer, R.W. and Chipman, J.W., in “Remote Sensing and Image Interpretation” 5th ed. Wiley, 2004

[2] Li Deng and Dong Yu “Deep Learning: methods and applications” by Microsoft research [Online] available at: http://research.microsoft.com/pubs/209355/NOW-Book-RevisedFeb2014-online.pdf

[3] McCulloch, Warren; Walter Pitts, “A Logical Calculus of Ideas Immanent in Nervous Activity”, Bulletin of Mathematical Biophysics 5 (4): 115–133(1943)

[4] An introduction to convolutional neural networks [Online]available at:http://white.stanford.edu/teach/index.php/An_Introduction_to _Convolutional_Neural_Networks

[5] Hubel, D. and Wiesel, T. (1968). Receptive fields and functional architecture of monkey striate cortex. Journal of Physiology (London), 195, 215–243C. J. Kaufman, Rocky Mountain Research Laboratories, Boulder, Colo., personal communication, 1992. (Personal communication)

[6] Yann LeCun, Leon Bottou, Yodhua Bengio and Patrick Haffner, “Gradient-Based Learning Applied to Document Recognition”, Proc. Of IEEE, November 1998.

[7] S. L. Phung and A. Bouzerdoum,”MATLAB library for convolutional neural network,” Technical Report, ICT Research Institute, Visual and Audio Signal Processing Laboratory, University of Wollongong. Available at: http://www.uow.edu.au/˜phung

[8] Tutorial on deep learning [Online] available at : http://deeplearning.net/tutorial/lenet.html

[9] Adelson, Edward H., Charles H. Anderson, James R. Bergen, Peter J. Burt, and Joan M. Ogden. “Pyramid methods in image processing.” RCA engineer 29, no. 6 (1984):33-41.

[10] M. Riedmiller and H. Braun, “A direct adaptive method of faster backpropagation learning: The rprop algorithm”, in IEEE International Conference on Neural Networks, San Francisco, 1993, pp. 586– 591.