**GOKARAJU RANGARAJU INSTITUTE OF ENGINEERING AND TECHNOLOGY**
**Department of Computer Science and Engineering**

## Real-time Research Project/Societal Related Project

# Real-time Pothole Detection System: Mitigating Road Hazards through Image Processing

Under the esteemed Guidance of
***- Dr. P. Chandra Sekhar Reddy***
***Professor***

**Team Members**
*Sai Priya-22241A0567*
*Anushka Kumar-22241A0570*
*K. Sannitha-22241A0586*
*Surugu Akshaya-22241A05C1*
*Sneha Padhi-22241A05C2*

# Contents

# Abstract

This project presents the development of a pothole detection system that uses dashcam footage and employs Convolutional Neural Network (CNN) models along with computer vision techniques to alert drivers when there is an upcoming pothole. Potholes are still a problem on roads around the world. These road defects are a serious threat to road safety, leading to vehicle breakdowns and accidents. Current pothole detection methods are manual and time-consuming, often hampering remediation strategies. The proposed system aims to enhance road safety by providing real-time warnings to drivers, enabling them to take appropriate actions on time.

The system utilizes a dashcam mounted on the vehicle to continuously capture video footage of the road ahead. This footage is processed using a CNN model trained to accurately detect and classify potholes in various road conditions and lighting scenarios. The detection is done using computer

vision algorithms which employs the trained CNN model. The driver is then alerted through auditory signals to allow immediate response.

The developed pothole detection application offers a practical solution for reducing vehicle damage and accidents caused by potholes, contributing to overall road safety and maintenance efficiency. This system can be integrated into modern vehicles, forming a crucial component of future transportation infrastructure.

# Literature Survey

| S.No. | Title of The Paper & Year | Authors | Methodology & Metrics | Datasets Used | Observed Shortcomings/ Gaps in The Paper |
|---|---|---|---|---|---|
| 1 | Pothole Detection System using deep learning | Samir Berjaoui, Anas Al-Shaghouri, Rami Alkhatib | 2D imaging & 3D reconstruction evaluated with Deep Learning (SSD-TF, YOLOv3, YOLOv4). YOLOv4 achieves best accuracy (mAP 85.39%) & speed (FPS) at 832x832 resolution. | Online database with 431 images. Combined: Lebanese roads (344) & web sources (312, mostly dashcam). | more complexity, less accuracy, less user-friendly, less robust to variations. |
| 2 | Detection and segmentation of concrete pothole based on Image Processing | Mingxing Gao, Xu Wang, Shoulin Zhu, Peng Guan | Texture, grayscale, morphology features + SVM for vehicle data. Evaluated with Recall, Precision, Accuracy, F1. | Images collected from agricultural and pastoral areas of Inner Mongolia, China. The exact number of images is not specified. | Image quality, Generalizability, External factors, Processing time, Feature selection, Sandy soil. |
| 3 | Real-time machine learning based approach for Pothole detection | Oche Alexander Egaji, Gareth Evans, Mark Graham Griffiths, Gregory Islas | Naive Bayes, Logistic Regression, KNN, SVM, Random Forest evaluated on k-fold CV with Accuracy, Precision, Recall, F1, AUC. | Phone/car sensor data (various routes) split into 2-second windows for training a classifier. | Data imbalance, real-world variations |

# Literature Survey

| S.No. | Title of The Paper & Year | Authors | Methodology & Metrics | Datasets Used | Observed Shortcomings/ Gaps in The Paper |
|---|---|---|---|---|---|
| 4 | Pothole detection system design with proximity sensor to provide motorcycle with warning system and increase road safety. | Hadistian Muhammad Hanif , Zener Sukra Lie, Winda Astuti , Sofyan Tan | Sensors (Proximity, ToF, Gyro, Hall Effect) + Arduino for distance, voltage, gyro, speed. | Roboflow and Kaggle Pothole Detection Dataset | Limited detection range, potential error in speed measurement, dependency on controlled conditions, complexity in implementation. |
| 5 | PotSpot: Participatory sensing-based monitoring system for pothole detection using deep learning | Susmita Patra, Asif Iqbal Middya, Sarbani Roy | Participatory sensing & custom CNN (TensorFlow Lite) achieve 97.5% accuracy on 3424 road images (InceptionV3, VGG16, VGG19). | 3424 road images dataset collected from real world scenarios | data quality issues, user participation, and scalability concerns |
| 6 | Application of various YOLO Models for Computer Vision-Based Real-Time Pothole Detection | Sung-Sik Park, Van-Than Tran, Dong-Eun Lee | YOLOv4/Tiny/v5 & AlexNet on 665 image dataset (mAP_0.5, Precision, Recall). | collection of 665 images captured at 720 × 720 resolution | Limited detection in certain conditions, manual labelling requirement, dependency on hardware resources, generalisation li mitations. |

# Literature Survey

| S.No. | Title of The Paper & Year | Authors | Methodology & Metrics | Datasets Used | Observed Shortcomings/ Gaps in The Paper |
|---|---|---|---|---|---|
| 7 | PotNet: Pothole detection for autonomous vehicle system using Convolutional Neural Network. | Deepak Kumar Dewangan, Satya Prakash Sahu | CNN (camera, Raspberry Pi) achieves 99.02% accuracy (Specificity 99.01%, Sensitivity 99.03%, Precision 98.03%, F1 98.33%). | public dataset named "Nienaber Potholes" accessed from Kaggle | Camera Calibration Needed, Distance Measurement Challenges |
| 8 | A Real-time Pothole Detection Based on Deep Learning Approach | Yeoh Keng Yik, Nurul Ezaila Alias, Yusmeeraz Yusof and Suhaila Isaak | YOLOv3 (Webcam, Multiprocessing) - mAP 65.05% (Precision 0.9%, Recall 0.45%). | 330 pothole images used to train the YOLOv3 model for real-time pothole detection | Calibration, Small Dataset, Low Power, Precision-Recall Tradeoff |
| 9 | DeepBus: Machine Learning based Real Time Pothole Detection system for smart transportation using IoT | Kashish Bansal, Kashish Mittal, Gautam Ahuja, Ashima Singh, Sukhpal Singh Gill | Accelerometer, Gyroscope, Random Forest tree-> Real-time map (86.8% Accuracy, 83.7% Precision, Rec, F1). | custom dataset was created for training the pothole detection system | Misclassifications, limited pothole severity detection, smartphone depen dency |

# Literature Survey

| S.No. | Title of The Paper & Year | Authors | Methodology & Metrics | Datasets Used | Observed Shortcomings/ Gaps in The Paper |
|---|---|---|---|---|---|
| 10 | Intelligent Real Time Pothole Detection and Warning System for Automobile Applications Based on IoT Technology | M.S. Kamalesh, Bharatiraja Chokkalingam, Jeevanantham Arumugam, Gomathy Sengottaiyan, Shanmugavadivel Subramani, Mansoor Ahmad Shah | Pi3, AWS, GSM/GPS, Accel, uC (ADXL345, Atmega328p) -> Image Processing, IoT. Tests across conditions. GPS in AWS (XYZ for potholes). | Not specifically mentioned | Processing power, network reliance, and potential device failures (ultrasonic). Optimal accuracy at 30 km/h. |
| 11 | YOLOv8-Based Visual Detection of Road Hazards: Potholes, Sewer Covers, and Manholes | Om Khare, Shubham Gandhi, Aditya Rahalkar, Sunil Mane | YOLOv8 (norm, aug, hypertune) on Roboflow's Intel "Pothole Detection" (mAP). | Roboflow Universe "Pothole Detection" dataset and trained on a dashboard camera perspective. | Requires diverse training data and further testing in various conditions. |
| 12 | Pothole Detection and Dimension Estimation System using Deep Learning (YOLO) and Image Processing | Pranjal A. Chitale, Kaustubh Y. Kekre, Hrishikesh R. Shenai, Ruhina Karani, Jay P. Gala | YOLO, Image Processing, Triangular Similarity (mAP, IoU) - Custom India/Foreign Road Dataset. | Custom Dataset (Indian and Foreign Roads) | Limited Dataset Size, Dependency on Camera Angle, Performance Variability with Environmental Factors |

# Literature Survey

| S.No. | Apps | Publishers | Methodology & Metrics | Observed Shortcomings/ Gaps in The Paper |
|---|---|---|---|---|
| 13 | Pothole Radar | JCL Apps | Centralized database stores all user marked potholes using gps and notify when a pothole is nearby | not user friendly, bad UI, only uses GPS to find potholes marked by users priorly not real-time detection. |
| 14 | RoadBounce Pothole Guard | Definitics | Mounts in car, detects potholes and clicks pictures, records GPS location & severity. Generates report after your drive | relying solely on phone sensors might not detect all potholes, especially smaller ones |
| 15 | Pothole Patrol | UMSEBENZI PATROL (PTY) LTD | Allows road users to report potholes in their area | Slow loading times, lack of response to issues, and excessive data collection |

# Requirements

- Software Requirements:
  - Programming language: Python
  - Integrated Development Environment (IDE): VSCode, Jupyter Notebook, Pycharm
  - Machine Learning Libraries(NumPy, OpenCV, TensorFlow, Matplotlib, SciPy, Keras, Pygame)
  - Development OS: Compatible with Windows, macOS, or Linux distributions.

- Hardware Requirements:
  - Processor: Intel Core i5 or higher
  - RAM: 8GB and above
  - Hard Disk: Minimum space of 100GB
  - Dashcam: A compatible dashcam (Webcam)
  - Audio Output: Output device to alert the driver
  - Connectivity: Internet Connection

# Proposed System

The proposed model for the pothole detection system is a convolutional neural network (CNN) designed to accurately identify potholes in real-time using dashcam footage. The model architecture balances complexity and efficiency to ensure detection accuracy while being capable of running on typical onboard hardware.

**1. Input Layer**

**Function:** Accepts resized images (e.g., 64x64 pixels) for standardized processing.

**Preprocessing:** Normalizes pixel values to enhance performance.

**2. Convolutional Layers**

**Purpose:** Extracts features such as edges, textures, and shapes.

**Details:** Utilizes multiple layers with ReLU activation functions for non-linearity.

**3. Pooling Layers**

**Function:** Downsamples feature maps using max-pooling to reduce dimensions and computational load.

**Benefit:** Captures dominant features, improving model invariance to translations and distortions.

**4. Dropout Layers**

**Purpose:** Prevents overfitting by randomly setting a fraction of input units to zero during training.

**Outcome:** Encourages learning of robust features.

**5. Fully Connected Layers**

**Role:** Aggregates features extracted by convolutional layers.

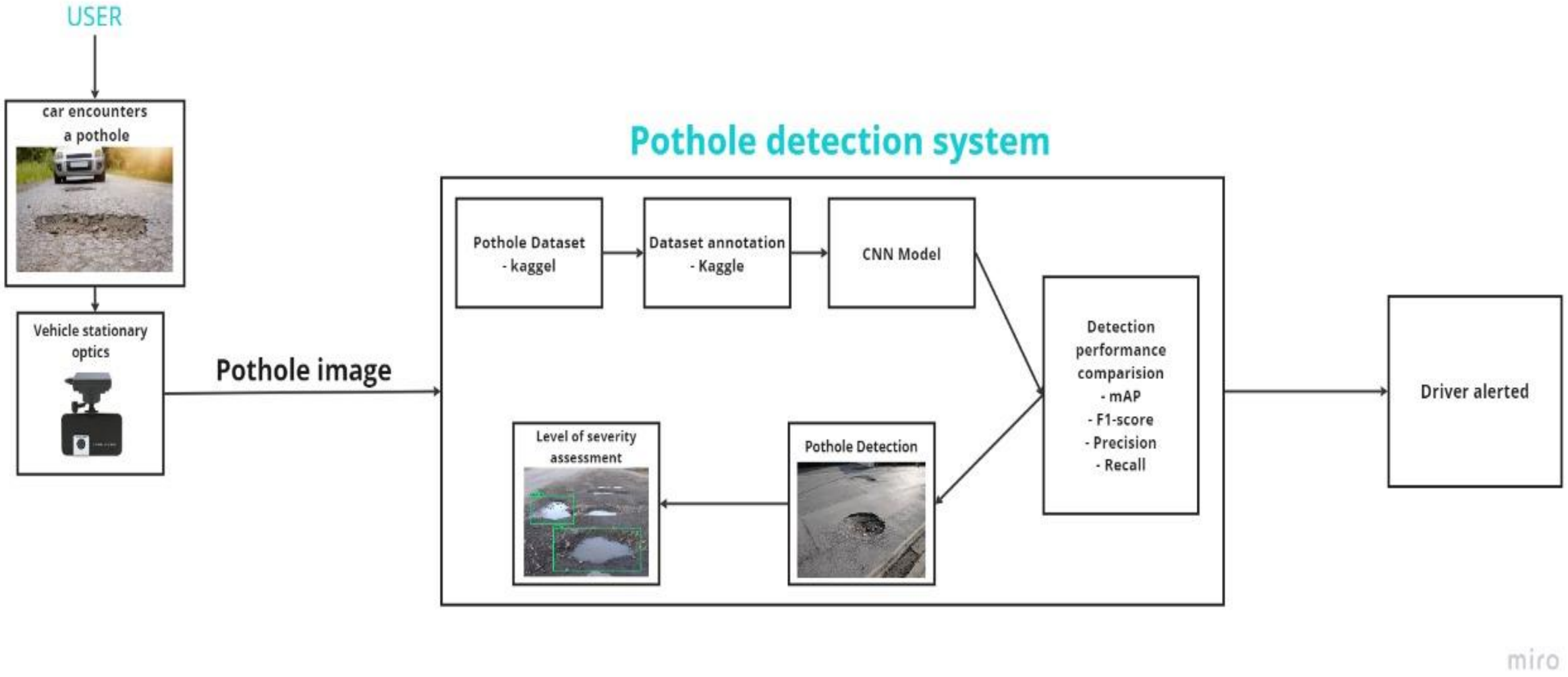
**Function:** Performs final classification based on learned features.

**6. Output Layer**

**Activation:** Uses softmax to provide class probabilities (pothole or normal road).

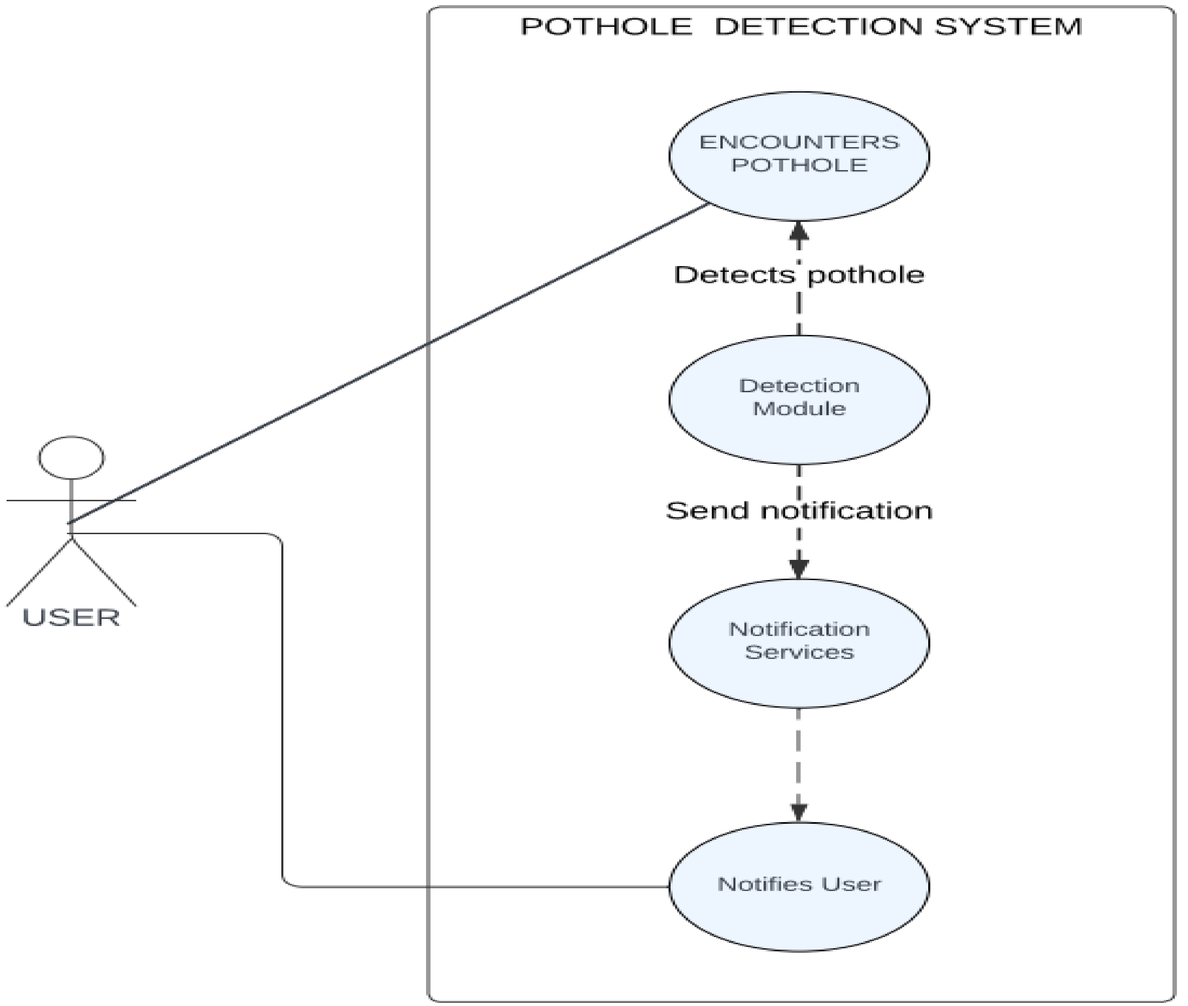
**Decision:** Selects the class with the highest probability as the prediction.

# System Architecture



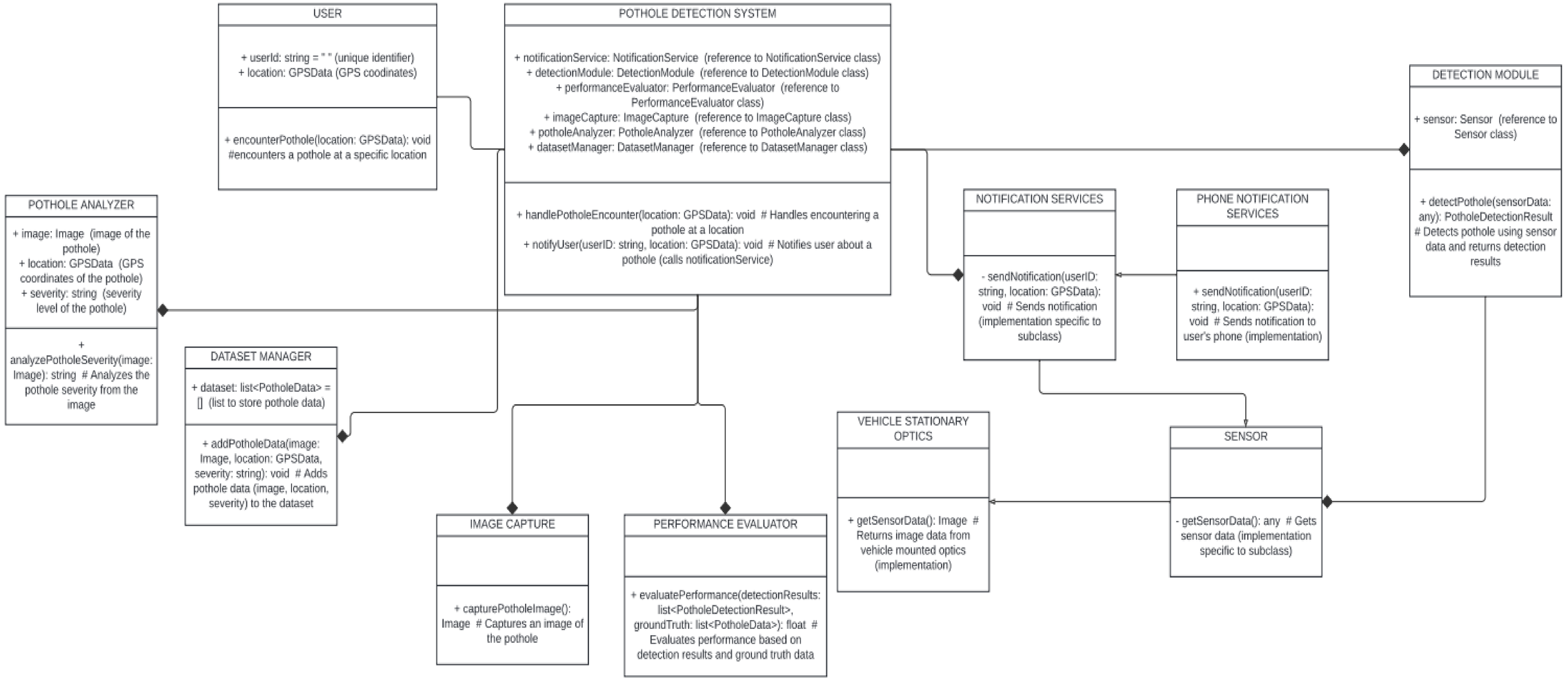
# Design – UML Diagrams

- Use Case Diagram



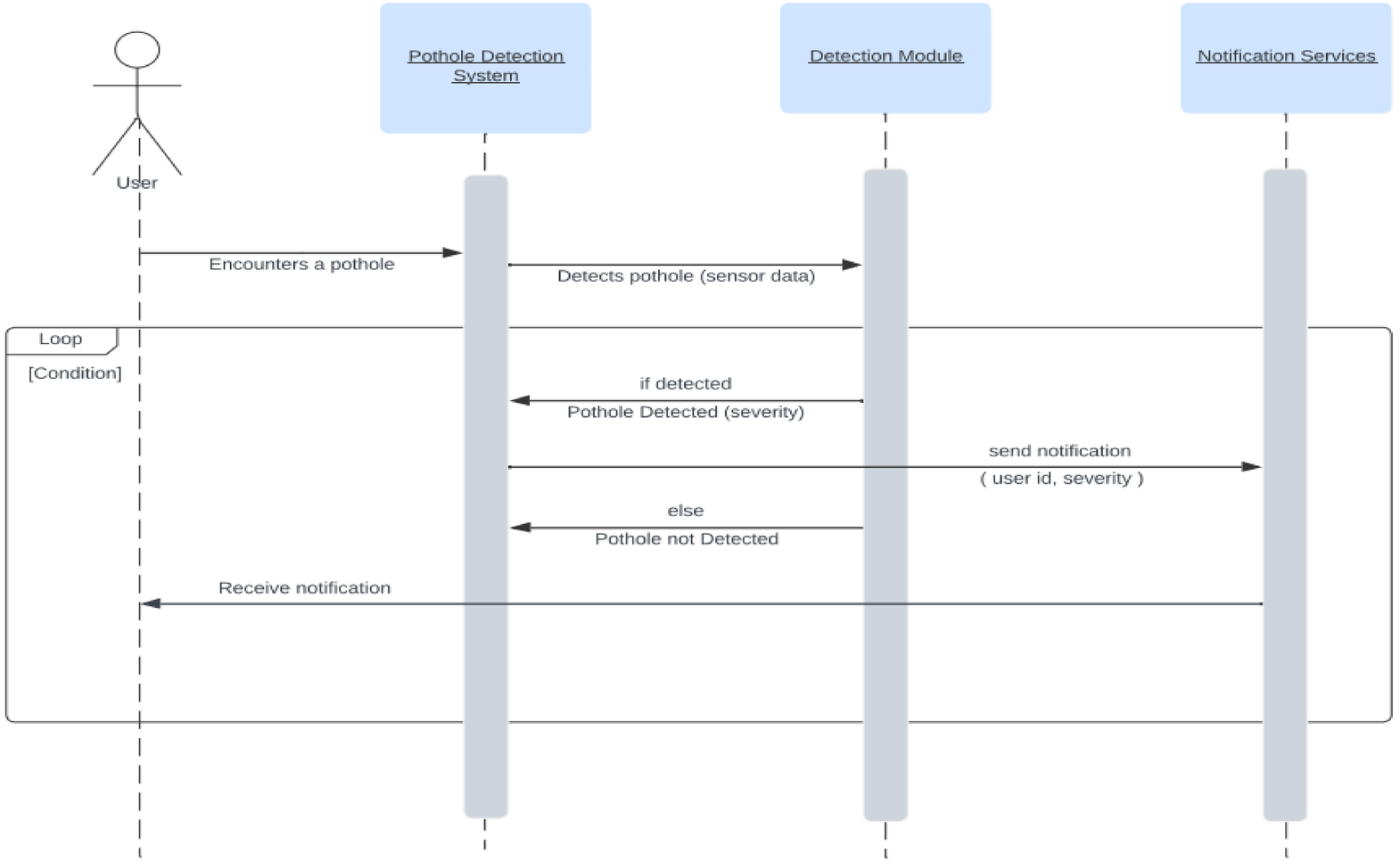
# Design – UML Diagrams

- Class Diagram



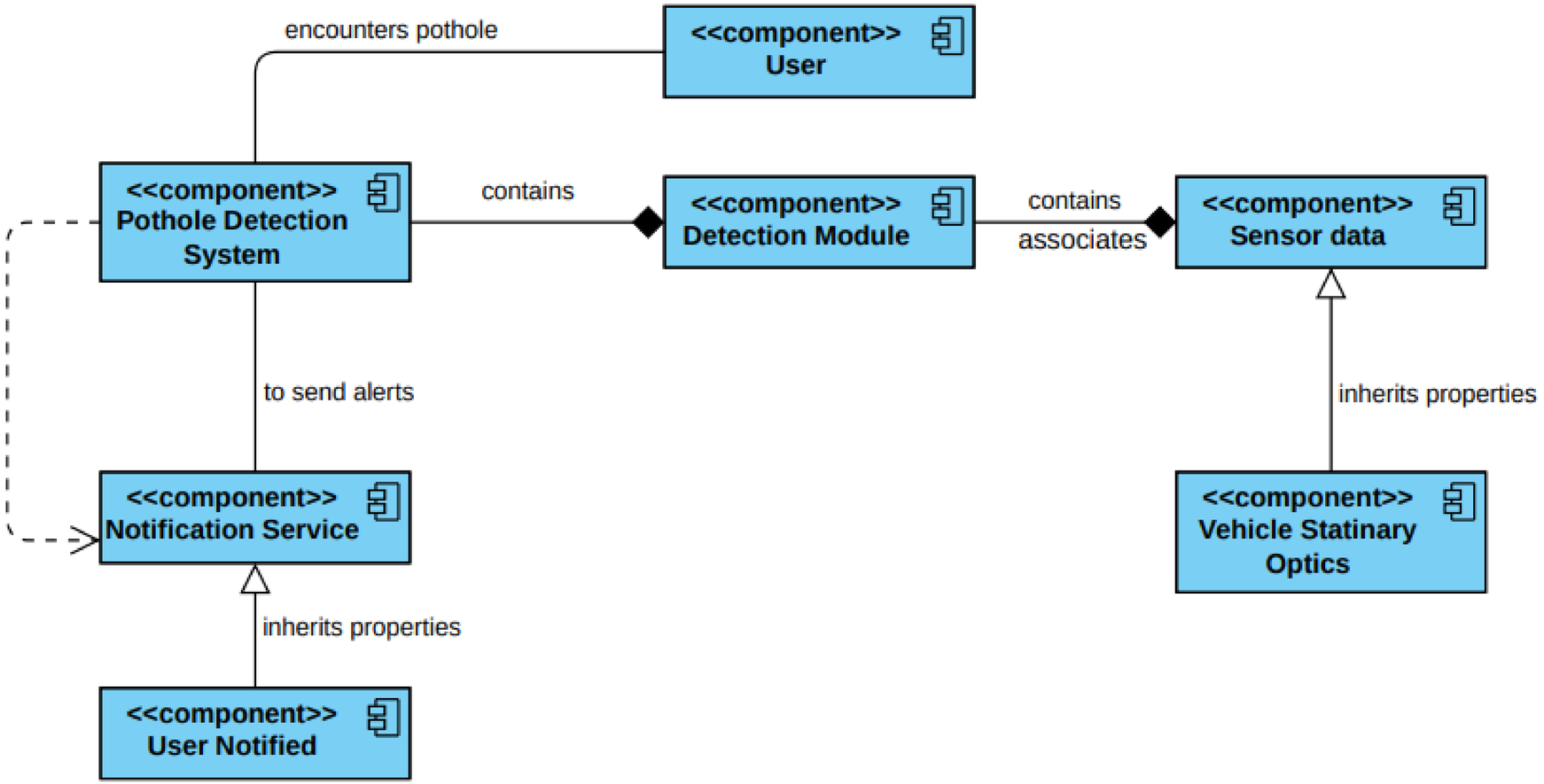
# Design – UML Diagrams

- Sequence Diagram



# Design – UML Diagrams

- Component Diagram



# Implementation – CNN Model

```
import os
import numpy as np
import cv2
import matplotlib.pyplot as plt
import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from keras.optimizers import Adam

import os
for dirname, _, filenames in os.walk("C:/Users/sneha padhi/OneDrive/Desktop/RTRP/archiv
e"):
    for filename in filenames:
        print(os.path.join(dirname, filename))

plt.imshow(cv2.imread("C:/Users/sneha padhi/OneDrive/Desktop/RTRP/archive/normal/110.jp
g"))

os.environ['TF_CPP_MIN_LOG_LEVEL'] = '2'

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

training_set = train_datagen.flow_from_directory('C:/Users/sneha padhi/OneDrive/Desktop/
RTRP/archive',
    target_size = (64, 64),
    batch_size = 32,
    class_mode = 'binary',
    subset="training")

validation_generator = train_datagen.flow_from_directory("C:/Users/sneha padhi/OneDrive/
Desktop/RTRP/archive",
    target_size=(64, 64),
    batch_size=32,
    class_mode='binary',
    subset='validation')
```

```
import tensorflow as tf

cnn = tf.keras.models.Sequential()
cnn.add(tf.keras.layers.Conv2D(filters=16, kernel_size=3, activation='relu', input_shape
=[64, 64, 3]))
cnn.add(tf.keras.layers.MaxPool2D(pool_size=2, strides=2))
cnn.add(tf.keras.layers.Conv2D(filters=32, kernel_size=3, activation='relu'))
cnn.add(tf.keras.layers.MaxPool2D(pool_size=2, strides=2))
cnn.add(tf.keras.layers.Conv2D(filters=64, kernel_size=3, activation='relu'))
cnn.add(tf.keras.layers.MaxPool2D(pool_size=2, strides=2))
cnn.add(tf.keras.layers.Flatten())

cnn.add(tf.keras.layers.Dense(units=128, activation='relu'))
cnn.add(tf.keras.layers.Dropout(0.4))
cnn.add(tf.keras.layers.Dense(units=1, activation='sigmoid'))

cnn.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

os.environ['TF_CPP_MIN_LOG_LEVEL'] = '2'
cnn.fit(x = training_set, validation_data = validation_generator, epochs = 100)
```

```
from keras.preprocessing import image
test_image = image.load_img("C:/Users/sneha padhi/OneDrive/Desktop/RTRP/My Dataset/test/
Plain/2.jpg", target_size = (64, 64))
plt.imshow(cv2.imread("C:/Users/sneha padhi/OneDrive/Desktop/RTRP/My Dataset/test/Plain/
2.jpg"))
test_image = image.img_to_array(test_image)
test_image = np.expand_dims(test_image, axis = 0)
result = cnn.predict(test_image)
training_set.class_indices
if result[0][0] == 1:
  prediction = 'pothole'
else:
  prediction = 'not pothole'


print(prediction)
```

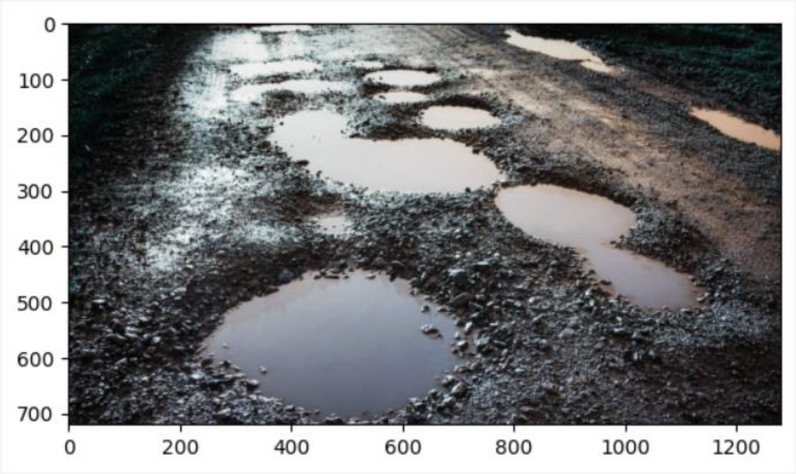
```
import cv2
import numpy as np
from tensorflow.keras.models import load_model
import tkinter as tk
from PIL import Image, ImageTk
import pygame

model = load_model('C:\\Users\\sneha padhi\\OneDrive\\Desktop\\RTRP\\model2.h5')

video_path = 'C:\\Users\\sneha padhi\\OneDrive\\Desktop\\RTRP\\DASH CAM 2016 01 29 (42 M
iles of Potholes).mp4'

root = tk.Tk()
root.title("Pothole Detection System")

canvas = tk.Canvas(root, width=1240, height=600)
canvas.pack()

label = tk.Label(root, text="", font=("Helvetica", 24))
label.pack()

pygame.init()
pygame.mixer.music.load('C:\\Users\\sneha padhi\\OneDrive\\Desktop\\RTRP\\beep.mp3')

cap = cv2.VideoCapture(video_path)

def process_frame():
    ret, frame = cap.read()
    if not ret:
        return

    img_resized = cv2.resize(frame, (64, 64))

    img_resized = img_resized.astype('float32') / 255.0
    img_resized = np.expand_dims(img_resized, axis=0)

    prediction_raw = model.predict(img_resized, verbose=0)
    maxArg = np.argmax(prediction_raw[0])
    pred = maxArg
    conf = prediction_raw[0][maxArg]
```

```
    if pred == 0 and conf >= 0.7:
        label.config(text="Pothole Detected!")
        pygame.mixer.music.play()
    else:
        label.config(text="")

    img = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
    img = Image.fromarray(img)
    img = ImageTk.PhotoImage(img)

    canvas.create_image(0, 0, image=img, anchor='nw')
    canvas.image = img

    root.after(1, process_frame)

process_frame()

root.mainloop()

cap.release()
cv2.destroyAllWindows()
```

# Results

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 14ms/step
not pothole
```


```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 51ms/step
pothole
```



# Conclusion and Future Scope

## Future Scope

The success of this project lays a solid foundation for further enhancements and future developments:

**Dataset Expansion**:

- Increase dataset size and diversity.
- Include more images of varying weather conditions, lighting scenarios, and road types.
- Enhance model accuracy and versatility.

**Model Optimization**:

- Experiment with different CNN architectures.
- Tune various hyperparameters.
- Explore advanced techniques like transfer learning for improved performance.

**Vehicle Integration**:

- Integrate into commercial vehicles as part of Advanced Driver-Assistance Systems (ADAS).
- Collaborate with automotive manufacturers.
- Test under real-world driving conditions.

**Extended Applications**:

- Extend framework to detect additional road anomalies (e.g., cracks, debris, construction zones).
- Increase overall utility and functionality.

# Conclusion and Future Scope

## Conclusion

This project successfully developed a real-time pothole detection system using convolutional neural networks (CNNs) and computer vision techniques, significantly enhancing road safety. The primary goal was to accurately detect potholes from dashcam footage and alert drivers in real-time, preventing potential vehicle damage and accidents.

We utilized a diverse dataset from Kaggle, containing over 700 images of various road conditions, which made the model robust and accurate. The CNN, implemented with TensorFlow and Keras, used convolutional layers for feature extraction, max-pooling layers for dimensionality reduction, and dense layers for classification, ensuring computational efficiency. Image preprocessing with techniques like rescaling, shearing, zooming, and horizontal flipping improved model performance on unseen data.

The model was trained and validated using categorical cross-entropy as the loss function and the Adam optimizer, ensuring efficient learning without overfitting. In real-time detection, video frames from dashcam footage were processed, and the model provided visual alerts and audio warnings via Pygame, enabling drivers to take corrective measures.

Testing showed high accuracy in pothole detection, validating the effectiveness of CNNs and computer vision for this application. The system's high functionality and reliability in real-time scenarios further confirmed its potential for enhancing road safety.