Pothole Detection and dimension Estimation using deep learning (YOLO) and image processing

Bhargav Ram Gowrisetty[1], Hareesh Reddy Emani[2], Vineeth Kumar Ganta[3], Doushik Chinthapudi[4]
[5] Shaik. Riyazuddien

[1-4] *Undergraduate Students, Department of ECE, Vasireddy Venkatadri Institute of Technology, Nambur, Guntur, Andhra Pradesh, India*
[5] *Associate Professor, Department of ECE, Vasireddy Venkatadri Institute of Technology, Nambur, Guntur, Andhra Pradesh, India*

ABSTRACT

The world is advancing towards an autonomous environment at a great pace and it has become a need of an hour, especially during the current pandemic situation. The pandemic has hindered the functioning of many sectors, one of them being Road development and maintenance. Creating a safe working environment for workers is a major concern of road maintenance during such difficult times. This can be achieved to some extent with the help of an autonomous system that will aim at reducing human dependency. In this paper, one of such systems, a pothole detection and dimension estimation, is proposed. The proposed system uses a Deep Learning based algorithm YOLO (You Only Look Once) for pothole detection. Further, an image processing based triangular similarity measure is used for pothole dimension estimation. The proposed system provides reasonably accurate results of both pothole detection and dimension estimation. The proposed system also helps in reducing the time required for road maintenance. The system uses a custom made dataset consisting of images of water-logged and dry potholes of various shapes and sizes.

Keywords: *Deep Learning, Pothole Detection, Convolutional neural networks(CNNs),YOLO(You Only Look Once)Open CV,Pandas,Numpy,Scikit,Tensor Flow,Keras,Pytorch*

---

1. INTRODUCTION

Potholes are a ubiquitous urban infrastructure challenge, causing damage to vehicles, posing safety risks to pedestrians, and incurring significant maintenance costs for municipalities. Pothole detection and dimension estimation have traditionally been manual and time-consuming tasks for maintenance crews. However, advancements in deep learning, particularly with convolutional neural networks (CNNs) and object detection frameworks like You Only Look Once (YOLO), coupled with image processing techniques, have revolutionized this process.

YOLO, known for its real-time object detection capabilities, is employed as the primary framework for identifying potholes within images or video streams. Image processing techniques complement the deep learning model by enhancing the quality of input images, extracting relevant features, and refining the output detections. These techniques may include preprocessing steps like noise reduction, contrast enhancement, and edge detection, which help improve the accuracy and reliability of pothole detection.

2. EXISTING POTHOLE DETECTION SYSTEM

Before the advent of pothole detection systems utilizing YOLO (You Only Look Once) and other deep learning techniques, traditional methods for detecting potholes typically relied on manual inspections or simple image processing algorithms. Here's an overview of some existing methods:

Manual Inspection: Historically, pothole detection primarily involved manual inspections by trained personnel. Maintenance crews or inspectors would visually inspect roads and highways for signs of damage, including potholes, cracks, and other defects.
Image Processing Techniques: Basic image techniques such as edge detection, thresholding, and morphological operations have been used for pothole detection. These methods analyze images captured by cameras mounted on vehicles or roadside installations.

2.1 Disadvantages in existing system
Many earlier systems struggled with accurately identifying potholes, often producing false positives or failing to detect smaller or less obvious potholes. This limited accuracy could lead to unnecessary maintenance or overlook actual road hazards. Some systems relied heavily on specific sensors or hardware installations on vehicles, roads, or infrastructure, making them costly to implement and maintain. Additionally, these sensors could be prone to damage or malfunction, impacting the reliability of the detection system.
Implementing and maintaining sophisticated pothole detection systems could be expensive, particularly for large-scale deployment across a road network. This cost factor often limited the widespread adoption of these technologies, especially in regions with limited resources. Previous systems may have suffered from significant processing time to analyze data and identify potholes accurately. This delay in detection and reporting could hinder timely maintenance efforts and increase the risk of accidents or road damage.

3. PROPOSED SYSTEM

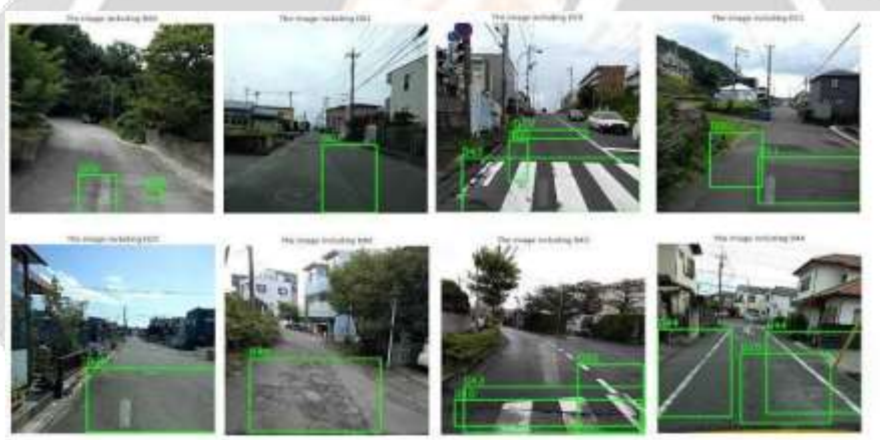

Figure1: Road damage detection and classifications.

The above Figure1 shows the examples of detected potholes on the road. It also indicates the pothole size and various parameters.

Potholes on roads pose a significant risk to both motorists and pedestrians, leading to accidents, vehicle damage, and discomfort. Automated detection of potholes using computer vision techniques can aid in timely repairs and maintenance, enhancing road safety and infrastructure management. This paper proposes a system based on YOLOv4, a state-of-the-art object detection algorithm, for the detection of potholes in road images. The proposed system aims to accurately identify potholes in real-time, enabling efficient maintenance scheduling and resource allocation. This paper outlines the architecture, training process, and evaluation metrics for the proposed pothole detection system using YOLOv4.

The proposed paper aims to contribute to the field of road infrastructure management by introducing a robust and efficient system for pothole detection using YOLOv4. By leveraging state-of-the-art deep learning techniques, the proposed system offers the potential for accurate and real-time detection of potholes, facilitating timely repairs and enhancing road safety.

3.1 Prerequisites & Environment:

Prerequisites and environment for pothole detection using YOLOv4 include:

Hardware Requirements:
GPU (Graphics Processing Unit): YOLOv4 training and inference are computationally intensive tasks, and having access to a GPU significantly accelerates the process.
Sufficient RAM: At least 16GB of RAM is recommended for efficient training.

Software Requirements:
Operating System: The system can run on various operating systems, including Linux, Windows, or macOS.
CUDA and CUDNN: Install CUDA and CUDNN libraries to enable GPU acceleration for deep learning tasks.
Python: YOLOv4 implementation typically uses Python programming language.
Deep Learning Framework: Choose a deep learning framework compatible with YOLOv4, such as TensorFlow
Image Processing Libraries: Install image processing libraries like OpenCV for handling image data.
Dataset: Collect a dataset containing images of roads with annotated potholes. Ensure that the dataset is diverse and representative of various road conditions, lighting conditions, and pothole sizes. Annotate the potholes in the images using bounding boxes or segmentation masks.

YOLOv4 Implementation:

- Download or clone the YOLOv4 repository from a reliable source.
- Set up the environment by installing necessary dependencies specified in the repository.
- Configure the YOLOv4 architecture according to the requirements of pothole detection.
- Customize configuration files, such as yolov4.cfg, to adjust parameters like input size, number of classes, and anchor boxes.

Preprocess the dataset according to the input format expected by YOLOv4.

Training:

- Initialize the weights of the YOLOv4 model with pre-trained weights (optional).
- Train the model on the annotated dataset using the chosen deep learning framework.
- Monitor training progress, adjust hyperparameters if necessary, and prevent overfitting.
- Save the trained weights and model checkpoints for future use.

Inference:
Perform inference on new road images using the trained YOLOv4 model. Post-process the detection results to filter out false positives and refine the predictions. Visualize the detected potholes on the road images along with confidence scores and bounding boxes.

Deployment:

- Integrate the trained YOLOv4 model into an application or system for real-time pothole detection.
- Optimize the inference process for efficiency, considering hardware constraints and performance requirements.
- Test the deployed system in real-world scenarios to validate its performance and accuracy.
- By fulfilling these prerequisites and setting up the appropriate environment, you can effectively implement pothole detection using YOLO.

3.2 Methodology

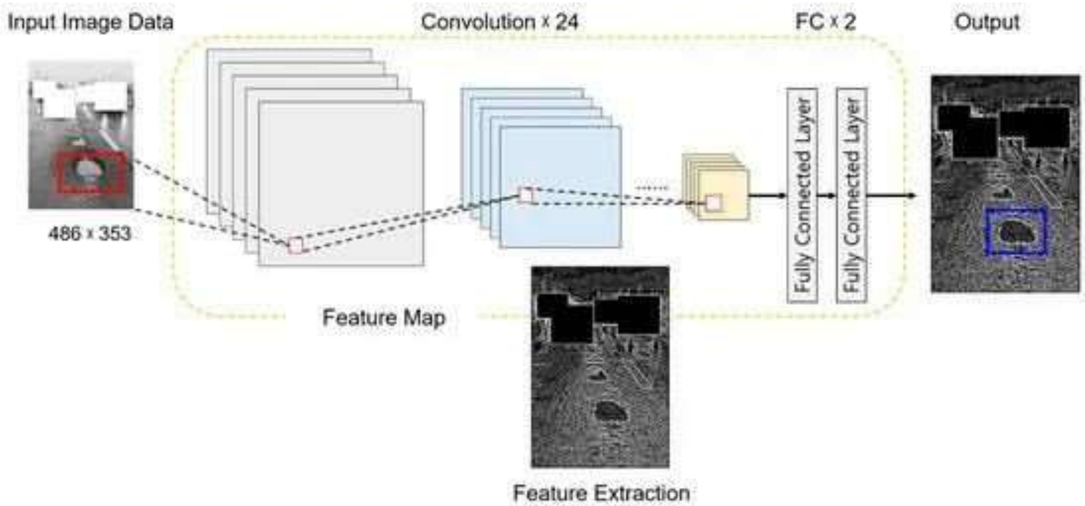

Figure2: Block diagram of Pothole Detection

The above block diagram illustrates the whole process of pothole detection. It tells us the every step involved in the pothole detection from taking the input data to processing the image and to getting the output data. It involves several steps.

An image datasets are taken as input parameter for the models shown below Figure 2, In data preprocessing model images are converted into a readable format and also images are resized to fit into input model. Dataset is divided into training and testing data. Creating advance CNN models with different layers and tuning the parameters. Evaluation metrics are calculated such as accuracy, precision, recall . Finally, will save all in one file as h5 and then creating the GUI application and selected the images and model for predicting the output based on the selected model the metrics and image classification is predicted. The implementation is done by using pretrained modules of deep learning algorithms which are used to predict the features taken for the datasets.

Implementation is done with 3 deep learning models those are Resnet50, vgg19, Inception-ResNet-v2. Dividing the data into training and testing then fit the model and evaluating to get the accuracy. The image datasets are used to compare the models and also the datasets whether for muddy roads or highway roads are giving the best result for the pretrained models. Vgg19 has given the best result because the architecture has 19 layers and also by hyper tunning the parameters this model work better compare to others models. gathering The dataset is collected from internet sources where images are high resolution for highway roads and muddy roads. Images are collected one dataset images are gathered from internet sources, muddy roads dataset and another dataset is taken from Kaggle[1] highway roads dataset. Images are classified into two labels Plain and Pothole. Total dataset contains around 1000 images/videos fig 2 shows some sample images from datasets.

Data Pre-processing. The image data taken as the input will have huge dataset with a lot of images .Resize the images into 256*256 to bring those images into a common format. All the images are made transformed through the label encoder. Defining the plain and pothole trained data as input for the models. Data Splitting Data is divided into two categories: training and testing. We must divide the entire data set into proportions. 65 percent will most likely be used for training, while 35 percent will be used for testing for highway road dataset and for muddy road dataset 60 percentage data for training and 40 percentage data for testing has taken . Testing dataset is a subset of the training dataset that is used to test learned models. Advance CNN models Deep learning models are used to predict the accuracy of the features which are taken to analyze. The models are Resnet50, InceptionResnetV2 and VGG19.First, we have selected each model one by one and trained the model using training data with that model. After successful training, the trained model is saved as h5.Then the prediction is done with each model and then it is evaluated with test dataset for accuracy. Architecture of models The models consists of five phases each with a convolution and Identity block. Each convolution blocks includes 3 convolution layers and each identity block comprises three convolution layers. Using Flask model we created a user interface which include the space for the giving input data for prediction and output. The web app will work as the font end part of the project. The training, testing and prediction will be run in the backend. The input image is selected to predict the output based on the selected models the predicted accuracy will be displayed with the classifying type.

4. RESULT:

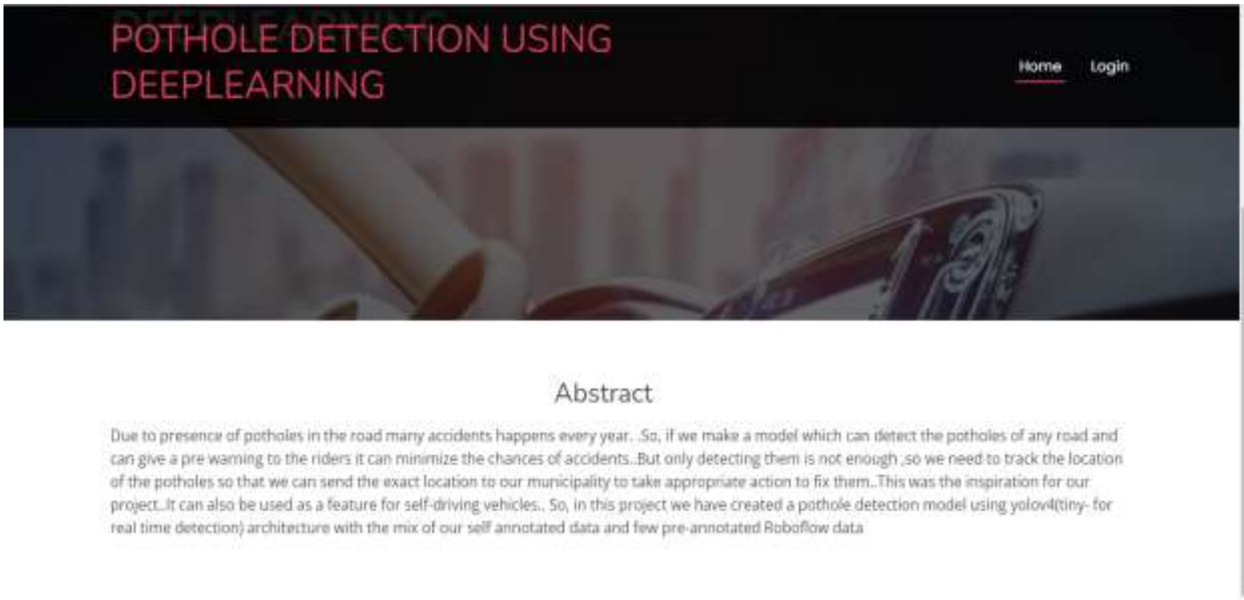

Figure3: Web Page

The above Figure3 is a home webpage which consists of abstract of the project and also a button to redirect to the login page.By running the main file we will get the http link for this page.If we copy paste the link in the browser we will redirect to the home page.This page is the beginning page of the pothole detection system.

Figure4:Login Web page

The above figure4 is a login web page. If we provide the correct credentials it will redirect to the main project.After providing correct credentials we will redirect to image lane detection page.In that page where we will give road image as input and we will get the lane detected image.

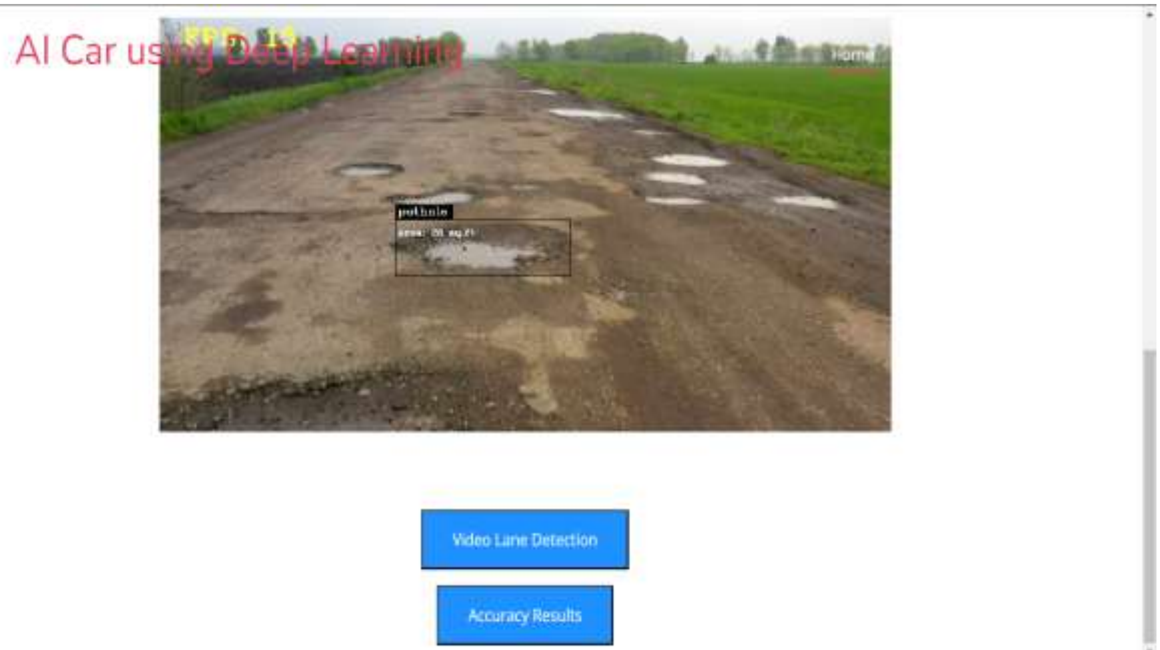

Figure5: Pothole Detected Image

The above figure5 is the pothole detected image which shows the area of the pothole and FPS of the system.During the detection of the pothole the system will alerts the driver by generating the beep sound.We can also perfom lane detection by using this system.

Figure6:Accuracy Plot

The figure6 is the overall accuracy plot of the project. It provides testing accuracy and training accuracy of the project.The plot is between the epoches and the accuracy of the system.As the epoches increases the accuracy of the system also increases.The training accuracy is represented in blue line and the testing accuracy is represented in red line.

5. CONCLUSIONS

Decision of using YOLO V4 was great because the biggest advantage of using YOLO is its superb speed – it's incredibly fast and can process 45 frames per second. YOLO also understands generalized object representation. It is one of the best object detection algorithms, with a performance that is comparable to that of the R-CNN algorithms. The system provides several benefits and can operate with less manpower. Hence, we have successfully completed the training and testing of our model using YOLO V4. The system successfully detects the potholes with a good accuracy of approx.97%. This work presented the state-of-the-art deep learning models (YOLO family and SSD-mobilenetv2) for real-time pothole detection leading towards the deployment on edge devices. Although, YOLOv5 showed the highest mAP@0.5 of 97% among other models but exhibits miss-classification and no detection potholes at long distances.A more sophisticated solution with the help of the global position system (GPS) can detect and point out the location of pavement failures. This work can contribute to self-driving applications and the automation industry. This work can further be extended to detect other pavement distresses, road depressions, classify roads as per quality, and depth estimation of potholes. The accuracy limitations can also be resolved in the future by further modification and extension in the real-time deployment.

Therefore, we concluded the YOLOv4 as the best-fit pothole detection model for accuracy and Tiny-YOLOv4 as the best-fit pothole detection model for real-time pothole detection with 95% detection accuracy and 31.76 FPS.

6. REFERENCES

[1] R. Fan, M. Liu, Road damage detection based on unsupervised disparity map segmentation (2019), IEEE Transactions on Intelligent Transportation Systems

[2] S. Chen, et al., 3D LiDAR scans for bridge damage evaluations (2012), Forensic Engineering 2012: Gateway to a Safer Tomorrow

[3] S. Sattar, et al., Road Surface Monitoring Using Smartphone Sensors: A Review (2018), Sensors (Basel, Switzerland)

[4] E. Buza, et al., Pothole detection with image processing and spectral clustering (2013), Proceedings of the 2nd International Conference on Information Technology and Computer Networks

[5] J. Singh, S. Shekhar, Road Damage Detection And Classification In Smartphone Captured Images Using Mask R-CNN (2018), arXiv preprint arXiv:1811.04535

[6] H. Maeda, et al., Road damage detection using deep neural networks with images captured through a smartphone (2018), Comput. Aided Civ. Infrastructure. Eng.

[7] O. Ronneberger, et al., U-net: Convolutional networks for biomedical image segmentation (2015), International Conference on Medical image computing and computer-assisted intervention