IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

StreetSafe: AI-Based Android Application for Pothole Detection

Kunal Chawla
Cse-AI &ML, Chandigarh University
Mathura, India

Vanshika Kaul
Cse- AI & ML, Chandigarh University
Gurgaon, India

Parth Thakkar
Cse- AI & ML, Chandigarh
UniversityNew Delhi, India

Shiv Sahil Soni
Cse- AI & ML, Chandigarh
UniversityNew Delhi, India

***Abstract*— Potholes are just a type of road imperfection that harms automobiles and jeopardizes drivers' safety. These anomalies in the road not only endangers the drivers but also the pedestrians. Numerous accidents and fatalities are brought up by potholes every year worldwide. Also, these potholes cause severe wear and tear upon vehicles, which is remedied by paying large sums of money to the automobile companies. These anomalies happen due to the usage of poor-quality construction materials and heavy rainfall. According to a report which was submitted by the Ministry of Road Transport and Highways research wing in New Delhi that almost 9500+ people died in 2017 due to these potholes. These mishaps can be avoided if the responsible authorities take strict actions to monitor and fix these potholes beforehand.**

Worldwide, maintaining roads properly is a very difficult task and a crucial issue. There is a huge demand for a robust system that can detect these anomalies beforehand and can alert drivers about them. This paper proposes a robust application that is fast and 97% accurate in detecting these anomalies in real-time without any buffering and which can be easily downloaded into the Car ecosystem via Google Play Store. This app uses effective Deep learning and machine learning techniques to detect these anomalies and alert its users beforehand.

Keywords: Application Programming Interface, Deep Learning, Convolution Neural Network, TensorFlow, Tflite, Neural Networks, Pothole detection.

I. INTRODUCTION

A pothole is a distinct kind of pavement distress that is brought on by severe weather and may be identified by a dip in the pavement surface with a minimum plan dimension of 150 mm. According to a recent WHO research, vehicle accidents are thought to be the cause of 13.5 lakh annual fatalities. Large collisions occur as a result of sloppy driving, bad weather, and other factors. Potholes and other irregularities in the road's surface are, by far, the most significant contributing element to these traffic incidents. The second-largest road network in the world is found in India. This indicates that the road system is vital to the Indian economy. According to reports, the road transport sector of India is growing at a rate of 10% as compared to overall annual GDP growth of 6%. Nowadays, the government is spending a lot of its resources in making these road networks but the main issue rises due to the maintenance of these roads. Unchecked potholes resulting from inadequate maintenance deteriorate the quality of roads, leading to deadly accidents and catastrophic injuries. Between 2013 and 2016, around 37000 people were injured and 11,836 people died as a result of potholes in India, according to government statistics. suffer with floods, heavy rainfalls and storms, etc. Though, we cannot eliminate the potholes but we certainly can reduce the number of road accidents by detecting the potholes, this can be done by providing an end-to-end system which can provide real time alerts to the users whenever the come in contact with any kind of potholes or bumps.

The four main phases make up the pothole detection process are: data collection, data preprocessing, feature extraction, and pothole categorization. In the data collection phase, we require enough data to analyze the potholes. In the data preprocessing step, we utilize the previously built data and perform techniques like data cleaning, data enrichment, data reduction, and data transformation so that it is ready for the third step. The feature extraction step is a process that is used to find out the patterns which can distinguish between potholes and non-potholes. By using the pothole detection algorithms, the pothole categorization phase aids in proving the existence of the potholes. Determining the nature of potholes requires careful data sensing and processing. The following is a summary of the general pothole detecting process. We can see this in **Figure 1**.

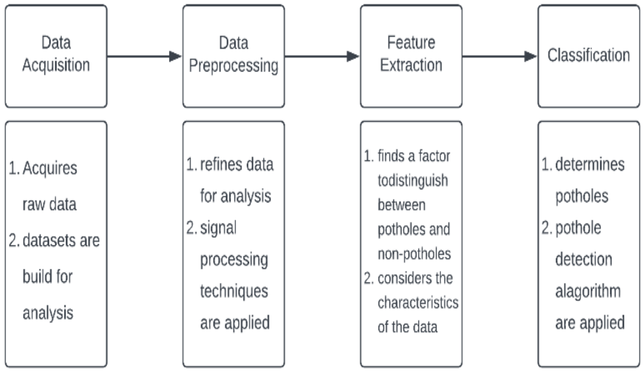


Fig 1: Phases of Pothole Detection

This research paper proposes an android application StreetSafe which is an end-to-end pothole detection application, which can be downloaded on any Android based device and which can help to combine all these processes and even save lives of innocent drivers and pedestrians. The methods proposed in this paper are proven to be 97% accurate in detecting these anomalies and alerting the user about these anomalies on the road.

II. LITERATURE SURVEY

Numerous researchers have been working on more precise methods of real-time pothole detection in recent years. In order to offer users an end-to-end solution, a number of strategies, including manual notification utilising mobile apps, Open-CV based techniques, sensor-based techniques, etc., have been created. These methods can be broadly categorised into three groups: 3D reconstruction-based methods, vibration-based methods, and methods based on vision. A approach based on vision identifies the existence of potholes using photos or videos as input data. The easiest and most economical technique is this one. Because it primarily bases its analysis on the vibration of the acceleration sensors, the vibration-based technique has the drawback of not providing the precise shape and depth of potholes. The stereo-vision-based 3D reconstruction-based approach estimates the volume and determines the precise shape of the potholes. This is the most accurate model but is not cost efficient to implement. The summary of these methods is shown in **Figure 2 and Table 1**.

Fig 2: Characteristics of automated pothole-detection methods

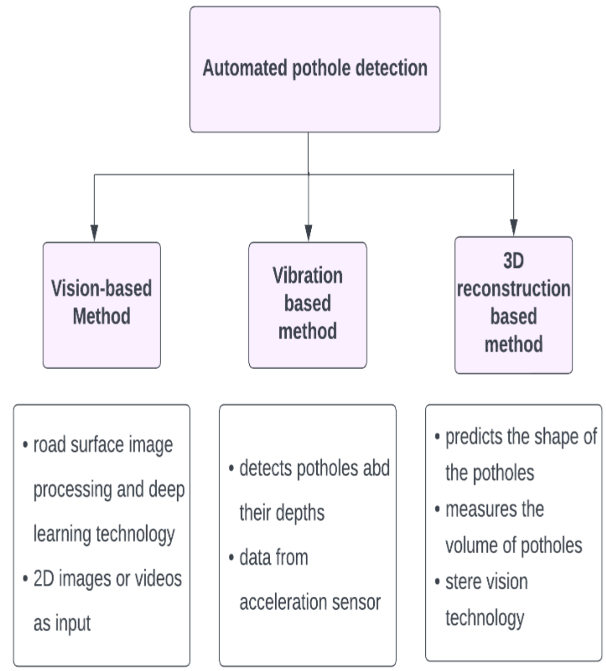


TABLE 1: The strengths and weakness of automated pothole-detection methods

Methods	Strengths	Weakness
Vision-based method	More affordable than 3D reconstruction based method Best for identifying potholes and estimating their size and shape	Cannot measure volume and depth of the potholes
Vibration-based method	Most cost-effective among the three methods Required small storage	Cannot provide the exact shape of pothole Real time detection cannot be done
3D reconstruction based method	Measures the shape of the potholes most accurately among the three methods	Most expensive among all the three methods

A lot of research work came into existence within a decade in which all the researchers are trying to make an end-to-end system which can help the drivers and can save the lives of innocent people. In the upcoming section, we have divided their work into three five mains sections and those are: Authors, detection techniques used by them, dataset they used to find out the potholes, the performance of those detection algorithms and the experimental circumstance they used in these pothole detection algorithms. After reviewing the data set section, we divided the dataset section into two parts total images used in Training the data set and total images used in testing the dataset. In the performance sector we used three metrics such as Accuracy, Precision and recall. This is clearly showcased in Table 2.

TABLE 2: A detailed comparison of the existing pothole detection techniques in terms if detection techniques, performance (accuracy, precision, recall), data set used and experimental circumstance.

S.NO.	Authors	Detection Techniques	Dataset	Performance	Experimental Circumstance
1.	Lim et al [1]	• Vision Based Pothole Detection • Training and testing is based on YOLO v2 • Uses a denser-grid model based on YOLO v2	Total: 1199 images Training : 996 images Testing:203 images	Accuracy: 67.7% Precision: 70.4% Recall: 74.9%	Used a GeForce GTX 1080
2.	Baek et al [2]	• Vision Based Pothole Detection • Feature Extraction is done using edge detection • Training and Testing is done using YOLO	Total: 13,376 images Training : 9364images Testing:2675 images	Accuracy: 77.8% Precision: 83.4% Recall: 72.9%	Used Windows 10, Intel Core i5-8440, 16GB RAM
3.	Hanshen Chen et al [3]	• Vision Based Pothole Detection • Training an testing is based on Location-aware CNN • Makes use of a component-primarily based type community(PCNN)	Total: 5676 images Training :4026 images Testing:1650 images	Accuracy: 95.0% Precision: 95.2% Recall: 92.0%	Used a GeForce GTX 1080
4.	Nhat-Duc Hoang [4]	• Vision Based Pothole Detection • Training an testing is based LS-SVM and ANN	Total: 200 images Training :160 images Testing: 40 images	Accuracy: 85.25% Precision: 92% Recall: 88%	N/A
5.	Azza Allouch [5]	• Vibration Based Pothole Detection • Data pre-processing done based on low pass filter • Feature Extraction is done based on Fourier transform and correlation • Classification is based on C4.05 decision tree, SVM and Naïve Bayes	Total: 2000 images Training : N/A Testing: N/A	Accuracy: 95.5% Precision: 98.5% Recall: 95.3%	Used cell phone mounted on the auto, android-based app operating on telephone
6.	Chao Wu [6]	• Vibration-based totally Pothole Detection • Statistics pre-processing is finished the usage of sliding window. • Feature Extraction is done using Fourier remodel and correlation • Education and trying out is done primarily based on Linear Regression, SVM and Random woodland	Total: 4088 images Training :3061 images Testing:474 images	Accuracy: 95.2% Precision: 85.1% Recall: 73.4%	Used experimental automobile, a telephone placed at the again seat and android primarily based app.
7.	Amita Dhiman [7]	• 3-D-reconstruction based Pothole Detection • Makes use of single-body stereo vision based totally method • Transfer learning with masks R-CNN and YOLOv2	N/A	Accuracy: 55.0% Precision: 45.8% Recall: 69.0%	Used GeForce GTX 1080 GPU and TESLA K80 GPU
8.	Jinchao Guan [8]	• 3D-reconstruction Based Pothole Detection • Training an testing is based on Principal Component Analysis • Uses Pixel-level pavement detection.	N/A	Accuracy: 95.9% Precision: 96.3% Recall: 95.5%	A vehicle equipped with GoPro HERO8

III. INFERENCE FROM RELATED WORK

As show in Table 2, the studies, which are conducted presently, are not focusing over creating a Real- Time detection system, which can accurately detect these potholes .The above studies lack a lot in accuracy, precision and recall. When we are dealing with lives of people a small discrepancy in these parameters may lead to fatal consequences. There is a huge need for a system, which is robust, real time, and with greater accuracy, precision and recall. The present studies are majorly providing accuracy results based on stock images of their testing dataset, while the methodology proposed in this research paper uses real-time images from the videos. The accuracy provided in this research paper is purely based on real-time images from the camera sensors of the smart-phone or the car dash cameras itself. Therefore, it is safe to conclude that there is a scope for some other more accurate ways to detect the potholes.

IV. PROPOSED WORK

The proposed pothole detection method detects the potholes in real-time. The method uses common techniques like "CNN", "Tflite", "Tfrecord" and "TensorBoard", "MobileNet" and "SSD". The main merit of these methods combined is small training time with a very easy training process and high accuracy. The dataset, which is utilized to train the model, is fetched from Kaggle.com. The data then undergoes through the data preprocessing and feature extraction process so that it can be utilized by the model. The transformed data is then sent through the Region-Based Convolution Neural Network(R-CNN), Tensorflow models to generate an output which is then fetched into the mobile application "StreetSafe". The trained model is then used to test over real time camera inputs from the camera sensors of smart-phones and of cars. These potholes are detected and are highlighted using a red-bounding-box around the potholes and also tells the distance of the vehicle with the potholes. This application also alerts the user with an audio-alert so that the driver becomes aware of the pothole ahead and can take decisions accordingly.

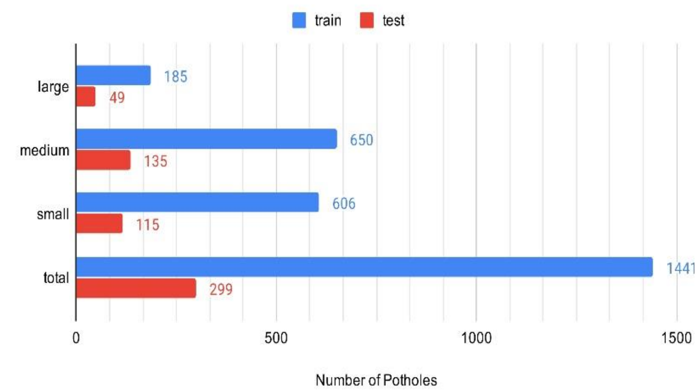
The data, which was sourced from Kaggle, is briefed under three major categories Small, Medium and Large. The pothole sizes from the occupied pixels are used to form these major groups. The amount of pixels was determined by scaling the images' longer sides to 300px.. The pixels v/s categories of potholes is represented in the below **Table 3.**

TABLE 3: Pixels Occupied v/s Categories of potholes

S.No.	Categories	Pixels Occupied
1.	Small	Area<= 32^2
2.	Medium	32^2< area<= 96^2
3.	Large	Area > 96^2

The distribution of the training and testing data over different categories is briefed under the Fig 3.

Fig 3: Training and Testing data brief



➔ Technologies Used

The application, which is proposed in this paper, utilizes languages such as Java and python for creating the application and generating a test dataset for the mobile application. The application is generated by using Android Studio (Dolphin) which is a mobile application development software powered by Google Inc. It provides many excellent features that enhances the productivity when building applications such as blended environments, flexible Gradle-based build system.

The target model was constructed using the transfer learning method. to identify the characteristics of pothole-prone areas. For the purpose of extracting features, a MobileNet model that had already been trained was used. The bounding boxes around potholes in several photos were predicted by the programme.

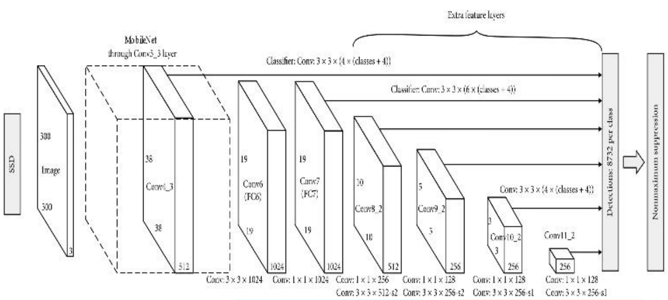
1. **CNN:** Deep learning algorithms like Convolution Neural Network are most effective for analysing and recognising images. Convolutional layers, pooling layers, and fully connected layers make up its three primary layers. The central elements of the CNN are the convolutional layers, where various filters are applied to extract features. The output is passed to the pooling layers, where the spatial dimensions are reduced to down-sample the data. The output is then passed through fully connected layers to make predictions.

2. **MobileNet:** It is a straightforward and effective convolutional neural network with high computational complexity that is employed in a variety of real-world applications, including object detection, fine-grained classifications, facial features, etc. It was created in 2017 by Google researchers. MobileNets uses 4.2 million parameters in total, which is a significant decrease from other CNN architectures. Also, it

offers the ability to regulate the model's size.

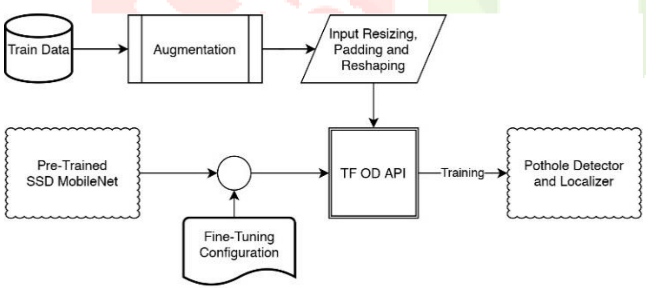
3. **SSD (Single Shot Multibox Detection):** To locate items in a video move, a deep learning version is used. it's miles a surely honest technique that has proven to be quite a success to this point in managing complex troubles. The spine version and the SSD Head are its two parts. As a function extractor, the backbone model is a pre-skilled picture categorization version. This backbone has layers known as SSD Head, whose outputs are interpreted as bounding containers. **Fig. 4** depicts the complete structure of the SSD.

Fig 4: Architecture of Single Shot MultiBox Detection



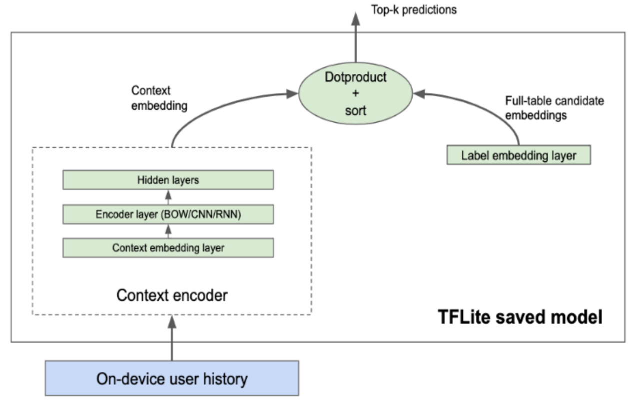
4. **Tensorflow API:** An open source library which was initially created by scientists and engineers working in the Google Brain Team within the Google Machine Intelligence research group. to undertake ML and Deep Neural Networks research. Tensorflow has APIs available in several languages. The Python API is at present the most easiest to use. In this project, the prepared dataset and certain fine-tuned parameters were used to train the Tensorflow API model. The API's process and how it is used in the project are shown in the figure below.

Fig 5: WorkFlow of TF OD API



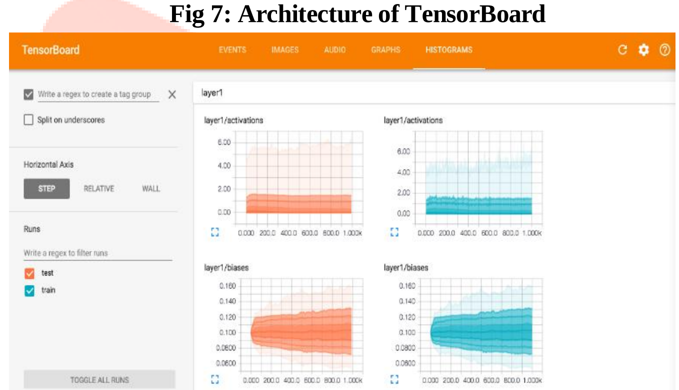
5. **Tflite:** Tensorflow lite is a hard and fast of equipment, which assist in on-device gadget mastering by helping the developers to run complex machine studying and deep gaining knowledge of strategies over their cellular, embedded and side gadgets. the important thing functions of Tflite is that is optimized for on-tool machine studying using five key constraints: latency, connectivity, and privacy, length and energy consumption. it is supported by different platforms along with Android, IOS, Embedded Linus and microcontrollers. It has a completely various language assist together with JAVA, fast, C++ and Python. The architecture of Tflite is showcased in the below **Fig 6**.

Fig 6: Architecture of Tflite



6. **TensorBoard:** TensorBoard is a visualization toolkit, that's used for machine mastering experimentation. It enables in tracking and visualizing metrics which includes loss and accuracy. It visualizes histograms of weights, biases as they exchange over the time. It shows text, audio and video statistics and lots greater. it is used in preparing the visualizations of accuracy, precision and don't forget of the version.

Fig 7: Architecture of TensorBoard



7. **Tfrecord:** all of the data is stored within the shape of tfrecords. It internally uses Protocol Buffers to serialize/deserialize the statistics and the shop them in bytes, as it will consume much less memory area to keep an adequate quantity of records. Protocol Buffers makes use of technology such as JSON & XML.

8. **Mediaplayer:** We have employed MediaPlayer, a component of the Android multimedia framework, which generates an audio warning if a pothole is detected.

V. PROJECTILE ANALYSIS

The work showcased in the research paper is highlighting the abilities of artificial neural networks and machine learning in the field of pothole detection. The mobile application developed has certain steps in order to provide accurate results. Here the input is the dataset, which is discussed earlier, and the output is the detection of the bumps and potholes.

Steps used to create Street Safe:

1. **Training the Model:** Education the version: The schooling dataset turned into first augmented in unique methods. The table under showcases the augmentation applied over the training dataset.

   a) Horizontal flip: it will reverse the pixels rows or column-sensible horizontally.
   b) Random Crop: it is approach in which we create a random subset of an original photograph.
   This allows in higher generalization of the version.
   c) Aspect Ratio Resizing: It's far used to have a easy grid of pics.
   d) Zero Padding: It's used
   to keep the original input length. that is specific in line with convolutional layer foundation.

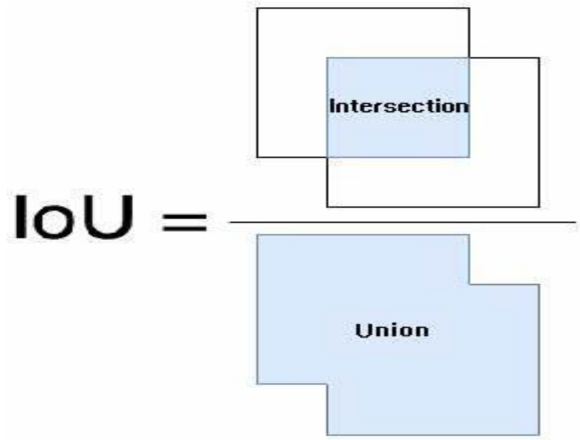
2. **Preparing a Pre-Trained Model:** Transfer learning approach is used in the generation of the model. Transfer Learning Technique that helps data scientists to benefit from the knowledge they have gained from previously used model for a similar task. This technique is applicable to many techniques such as deep learning models. Target of the model is to leverage pre-trained model's knowledge while performing a different task.
   In this mobile application, we used the Single Shot Multibox Detection and MobileNet v2 pre-trained model. In every pre-trained model, which is present in the Tensorflow Model Zoo we have a "pipeline.config" file, which is used for configuration file during the training and evaluation. Some of the changes were made to this file, which is showcased in the table below.

Configuration Options	Changes Made
Input-Shape	300,300,3
Num-Classes	1
Batch-Size	24
Min/Max Dimension	300
Initial-Learning-rates	0.005
Decay-Steps	6000
Decay-factor	0.85

3. **Run Training Process:** Tensorflow models fashioned the foundation for the training method and came up with the important python code. If the manner is interrupted in any way, it additionally offers the capability of saving the checkpoints. The scalar and picture values are saved in TensorBoard.
4. **Run Evaluation Process:** After the completion of training technique, we will pass ahead at the validation and evaluation dataset. Right here we applied the tensorflow models repository, which triggers the training tool every time it saves a checkpoint of the version. This feature allows to display the overall performance of the model at extraordinary stages.

5. **Evaluation Methods:** On this step, we evaluated the overall performance of the model. To take detected bounding box as true positive, different IoU (Intersection over Union) thresholds were taken into consideration. IoU is used to explain the extent of overlap of two packing containers. It's far immediately proportional to the location of overlap. it's far utilized in applications associated with object detection, in which a model is trained to model an output of a subject that suits perfectly around an object. The purpose of that is to improve the prediction of the devices.

$$\text{IOU} = \frac{Area\ of\ Intersection\ of\ two\ boxes}{Area\ of\ Union\ of\ two\ boxes}$$



10 IoU thresholds had been considered from 50% upto 95% with intervals of five% and these values have been then averaged to get the price for IoU. Taking the above IoU thresholds we have:

TP= Number of True Positive Predictions
FP= Number of False Positive Predictions
TN= Number of True Negative Predictions
FN= Number of False Negative Predictions

a) **Precision Calculation:** Precision is defined as the ratio of correctly classified positive samples divided by total number of positive samples(either correct or incorrect)

$$Precision = \frac{TP}{TP + FP}$$

Average precision is the area under the precision-recall curve.

$$AP = \int_0^1 p(r)dr$$

b) **Recall Calculation**: It is determined as the proportion of positively classed positive samples to all positively classified positive samples.

$$Recall = \frac{TP}{TP + FN}$$

Average Recall is the mean of the recall values of all individual classes.

$$AR = 1/N \sum_{i=1}^{N} ri$$

c) **Accuracy:** It is the ratio of correct prediction divided by correct prediction + incorrect prediction. If a model has high accuracy, we can infer that the model makes correct prediction most of the time.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

6. **Creating Application Layouts:** In this process we are creating the Application layouts using the .xml files. In addition, creating the classes, which are to be utilized while creating methods for the execution of the TensorFlow.

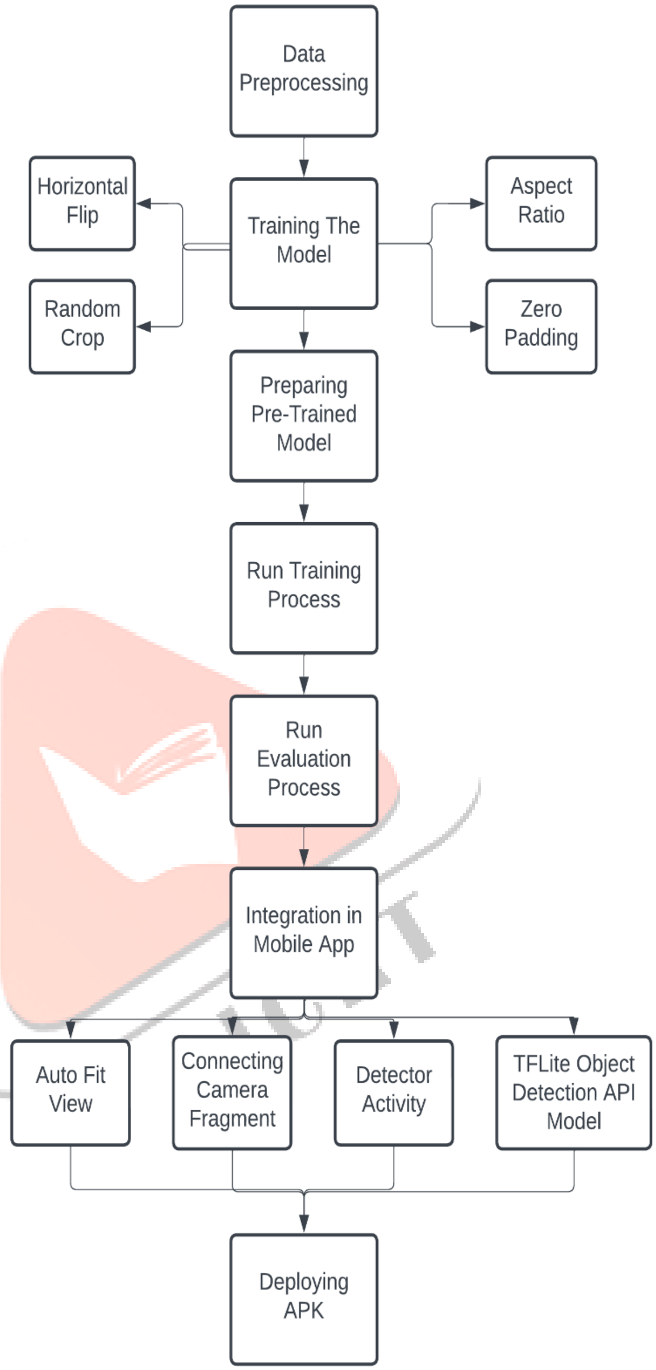
7. **Creating Camera Connection Fragments and Detector Activity:** In this step, we are connecting the xml files with the fragments of Camera. Here we are creating the optimal size of the red boxes that is to be showcased over the detected potholes. Detector Activity is used to import and detect the potholes by using the model, which was created in the previous steps.

8. **Creating TFLite Object Detection API Model in Android Studio:** The API model of the TFLite is made in this step by utilizing the training dataset prepared earlier. The file handles each type of errors using the try and except clauses and recognizes the images and applies the model over it to provide accurate results and sends information to the detector file to mark the region as the potholes by using the red bounding-boxes and alert the user.

9. **Building the Gradle and the Application:** The software is built using Gradle, an automation tool. The code is compiled, linked, and packaged during the construction process. In this mobile Application, we have preferred a stable SDK 29 for which stable TensorFlow API are present and can be utilized easily. These sdk provides the android studio about the total number of features to be used by the app.

10. **Running the App:** The final step is to check that the application is running without any error and is able to detect the potholes without any delay. If any error occur then we are required to debug it and find solution for the same. After successful run over a demo smart-phone, we can use its Apk from anywhere.

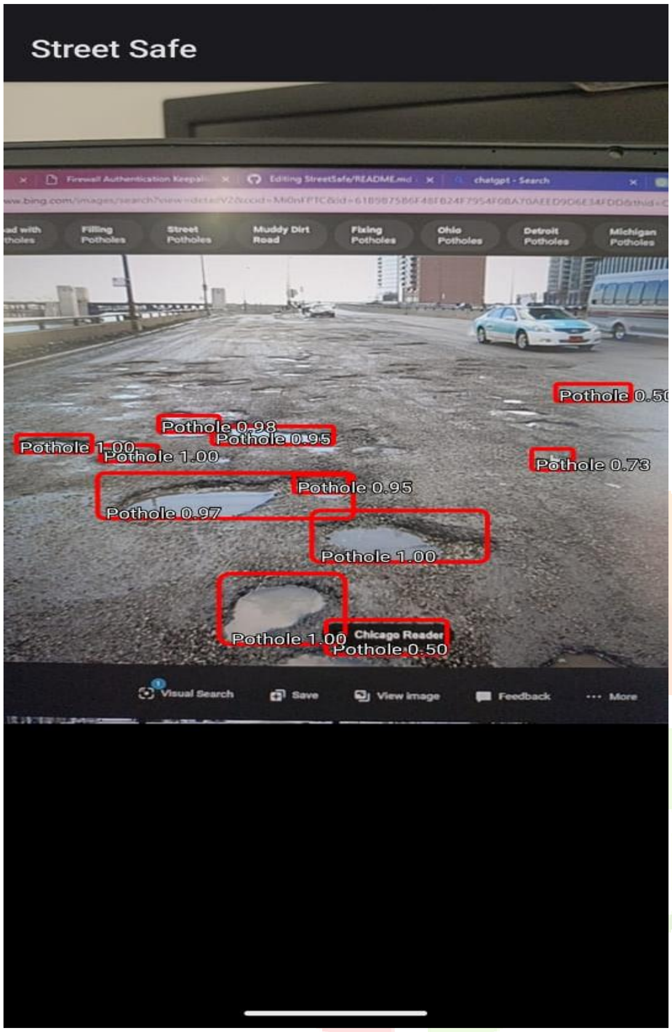
Fig 8: Data Flow Diagram



VI. RESULTS & DISCUSSION

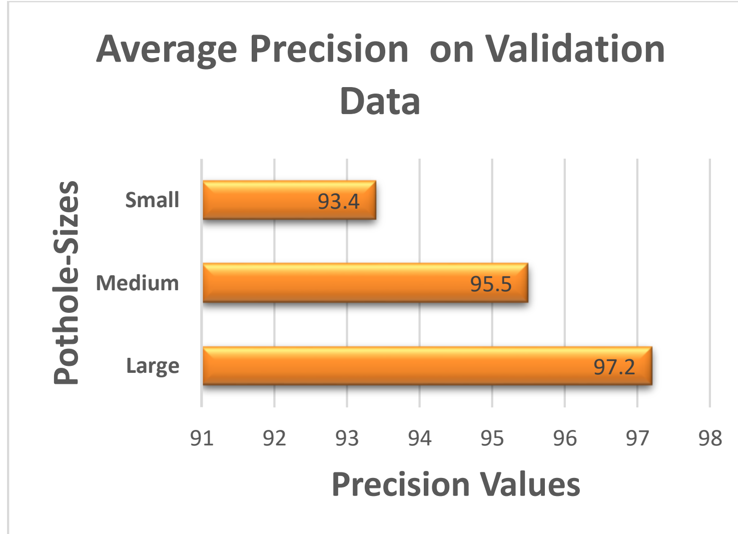
The proposed mobile application and model, which detects the potholes, is shown below in Fig 9.

Fig 9: Screen Shot of Mobile Application StreetSafe detecting Potholes



The above Screen Shot of the application showcases the ability of the applications to detect the potholes accurately and telling the user how close is the vehicle to the potholes. The training and evaluation process was run for more than 10,000 steps. The evaluation process was run with help of TensorFlow repositories. The performance of the model is considered as Average Precision considering IoU thresholds as the threshold for True Positive. The Average precision came out as 95.7% which is far greater than any other model present till now which is showcased in Chart 1. Accuracy of the model came out to be approx. 97.3%, which is by far the greatest considering previous deep learning models developed.

Chart 1: Precision V/s Pothole Sizes



It is clearly depicted that the model performs better on large potholes when compared to small potholes. The precision and accuracy values are good, which is unusual for a lightweight model that is based on MobileNet and SSD and designed to work with both high-end and low-end smartphone ecosystems as well as automotive ecosystems.

VII. CONCLUSION

One of the most crucial responsibilities for coming up with maintenance and rehabilitation plans is spotting potholes. The thousands of lives lost in accidents caused by potholes can be prevented by spotting them early and avoiding them. The work required to manage potholes is reduced thanks to this automated technology, which greatly benefits road maintenance. It is reasonable to anticipate that study in this area will continue in light of current technological trends.

The end-to-end Android application we have described in this research can be used to detect potholes in real time and send out warning messages to help the driver avoid them. This application is able to detect potholes more accurately as compared to any computer-vision based model. A convolutional neural network named MobileNet and Single Shot MultiBox Detector was used in a tuned manner to detect potholes. The potholes detection model presented in this paper can be utilized in many fields such as road management with the intelligent transportation systems. It can be used in making preventive road maintenance policies by sharing the data collected from the application. It can help in enhancing the suspension system of the vehicles and furthermore, it can largely contribute in the performance of autonomous-driving technology.

VIII. FUTURE WORK

When dealing with lives of people we require a system, which is 100% accurate. In future, we will be working over a more accurate and accomplished model, which can predict more accurately, is more robust and is more intelligent. In the near future, we will be introducing new features to the application such as a map, which can locate and tell the user where the potholes are present on the road. A layout where we can spread awareness about the traffic rules. A consumer centric user interface, which is highly interactive and is fun to use. We will be creating a system through which we can store the information collected by the users of our application and can share it will Government officials so that they can rehabilitate it and provide smooth road experience to the citizens.

IX. REFERENCES

[1] Lim, K.S.; Kwon, J.W. Detection of Potholes Using a Deep Convolutional Neural Network. J. Univers. Comput. Sci. 2018, 24,

[2] Baek, J.-W.; Chung, K. Pothole Classification Model Using Edge Detection in Road Image. Appl. Sci. 2020, 10, 6662.

[3] Chen, H.; Yao, M.; Gu, Q. Pothole detection using location-aware convolutional neural networks. Int. J. Mach. Learn. Cybern. 2020,11, 599–911.

[4] Hoang, N.-D. An Artificial Intelligence Method for Asphalt Pavement Pothole Detection Using Least Squares Support Vector Machine and Neural Network with Steerable Filter-Based Feature Extraction. Adv. Civ. Eng. 2018, 2018, 7419058.

[5] Allouch, A.; Koubâa, A.; Abbes, T.; Ammar, A. RoadSense: Smartphone Application to Estimate Road Conditions Using Accelerometer and Gyroscope. IEEE Sens. J. 2017, 17, 4231–4238.

[6] Wu, C.; Wang, Z.; Hu, S.; Lepine, J.; Na, X.; Ainalis, D.; Stettler, M. An Automated Machine-Learning Approach for Road Pothole Detection Using Smartphone Sensor Data. Sensors 2020, 20, 5564.

[7] Dhiman, A.; Klette, R. Pothole Detection Using Computer Vision and Learning. IEEE Trans. Intell. Transp. Syst. 2020, 21,3536–3550.

[8] Guan, J.; Yang, X.; Ding, L.; Cheng, X.; Lee, C.S.; Jin, C. Automated pixel-level pavement distress detection based on stereo vision and deep learning. Autom. Constr. 2021, 129, 103788.

[9] Maxwell, A.E.; Warner, T.A.; Guillén, L.A. Accuracy Assessment in Convolutional Neural Network-Based Deep Learning Remote Sensing Studies—Part 1: Literature Review. Remote Sens. 2021, 13, 2450.

[10] B. X. Yu, and X. Yu, Vibration-based system for pavement condition evaluation, In Proceedings of the 9th International Conference on Applications of Advanced Technology in Transportation (2006), 183-189.

[11] K. De Zoysa, C. Keppitiyagama, G. P. Seneviratne, and W.W.A.T. Shihan, A public transport system based sensor network for road surface condition monitoring, In Proceedings of Workshop on Networked Systems for Developing Regions (2007), 1-6.

[12] J. Erikson, L. Girod, and B. Hull, The pothole patrol: using a mobile sensor network for road surface monitoring, In Proceedings of the 6th International Conference on Mobile Systems, Applications, and Services (2008), 29-39.

[13] A. Mednis, G. Strazdins, R. Zviedris, G. Kanonirs, and L. Selavo, Real time pothole detection using Android smartphones with accelerometers, In Proceedings of the International Conference on Distributed Computing in Sensor Systems and Workshops (2001), 1-6.

[14] K. C. P. Wang, Challenges and feasibility for comprehensive automated survey of pavement conditions, In Proceedings of 8th International Conference on Applications of Advanced Technologies in Transportation Engineering (2004), 531-536.

[15] K.T. Chang, J. R. Chang, and J. K. Liu, Detection of pavement distress using 3D laser scanning technology, In Proceedings of the ASCE International Conference on Computing in Civil Engineering (2005), 1-11.

[16] Z. Hou, K. C. P. Wang, and W. Gong, Experimentation of 3D pavement imaging through stereovision, In Proceedings of the International Conference on Transportation Engineering (2007), 376-381.

[17] Ai D, Jiang G, Kei LS, Li C (2018) Automatic pixel-level pavement crack detection using information of multi-scale neighborhoods. IEEE Access 6:24,452–24,463

[18] Albawi S, Mohammed TA, Al-Zawi S (2017) Understanding of a convolutional neural network. In: 2017 International conference on engineering and technology (ICET). IEEE, pp 1–6

[19] Ali A, Zhu Y, Zakarya M (2021) A data aggregation based approach to exploit dynamic spatio-temporal correlations for citywide crowd flows prediction in fog computing. Multimed Tools Appl:1–33

[20] All about Android Platforms. [Online] Available: https://developer.android.com/about

[21] Anaissi A, Khoa NLD, Rakotoarivelo T, Alamdari MM, Wang Y (2019) Smart pothole detection system using vehicle-mounted sensors and machine learning. J Civ Struct Health Monitor 9(1):91–102

[22] Anand S, Gupta S, Darbari V, Kohli S (2018) Crack-pot: Autonomous road crack and pothole detection. In: 2018 Digital image computing: Techniques and applications (DICTA). IEEE, pp 1–6

[23] Arya D, Maeda H, Ghosh SK, Toshniwal D, Mraz A, Kashiyama T, Sekimoto Y (2020) Transfer learning-based road damage detection for multiple countries. arXiv:2008.13101

[24] Bhatia Y, Rai R, Gupta V, Aggarwal N, Akula A et al (2019) Convolutional neural networks based potholes detection using thermal imaging Journal of King Saud University-Computer and Information Sciences