VISVESVARAYA TECHNOLOGICAL UNIVERSITY

Jnana Sangama, Belgaum-590018

A PROJECT REPORT **(15CSP85)** ON

Use of AI for preventive road maintenance

Submitted in Partial fulfillment of the Requirements for the Degree of

Bachelor of Engineering in Computer Science & Engineering

By

Pavan Vamsi Tadikonda (1CR15CS107)

Mir Owais (1CR15CS096)

Shubham Kumar (1CR15CS096)

Under the Guidance of,

N. Sreedevi

Designation, Dept. of CSE

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CMR INSTITUTE OF TECHNOLOGY

#132, AECS LAYOUT, IT PARK ROAD, KUNDALAHALLI, BANGALORE-560037

CMR INSTITUTE OF TECHNOLOGY

#132, AECS LAYOUT, IT PARK ROAD, KUNDALAHALLI, BANGALORE-560037

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

Certified that the project work entitled **"Use of AI for preventive road maintenance"** carried out by **Mr Pavan Vamsi Tadikonda**, USN **1CR15CS107**, **Mr Mir Owais**, USN **1CR15CS150**, **Mr Shubham Kumar**, USN **1CR15CS150,** bonafide students of CMR Institute of Technology, in partial fulfillment for the award of **Bachelor of Engineering** in Computer Science and Engineering of the Visveswaraiah Technological University, Belgaum during the year 2019-2020. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the Report deposited in the departmental library.

The project report has been approved as it satisfies the academic requirements in respect of Project work prescribed for the said Degree.

.

N.Sreedevi	**Dr. Prem Kumar Ramesh**	**Dr. Sanjay Jain**
Associate Professor	**Professor & Head**	**Principal**
Dept. of CSE, CMRIT	**Dept. of CSE, CMRIT**	**CMRIT**

DECLARATION

We, the students of Computer Science and Engineering, CMR Institute of Technology, Bangalore declare that the work entitled "**Use of AI for preventive road maintenance**" has been successfully completed under the guidance of Prof. N.Sreedevi Computer Science and Engineering Department, CMR Institute of technology, Bangalore. This dissertation work is submitted in partial fulfillment of the requirements for the award of Degree of Bachelor of Engineering in Computer Science and Engineering during the academic year 2019 - 2020. Further the matter embodied in the project report has not been submitted previously by anybody for the award of any degree or diploma to any university.

Place:

Date:

Team members:

Pavan Vamsi Tadikonda (1CR15CS107) ____________________

Mir Owais (1CR15CS096) ____________________

Shubham Kumar (1CR15CS150) ____________________

ABSTRACT

Current method of surveying roads in India is a manual task. It is laborious and costly.
This leads to lesser surveying and lesser reporting of damages in roads, which in turn leads many accidents and deaths.
We aim to remove human intervention in the process of surveying roads and make the task easy and cheap. The goal is to develop a cheap and effective device that can automatically detect the damages in roads (such as potholes or abrasions).
The device will also be able to perform image recognition on the damages done and provide meaningful and reasonable estimate for the cost of repair. There will also be intelligent analysis capability.
we use machine learning techniques such as clustering and random forest regression to obtain estimates for future detected damages.
The project will also feature the ability to perform detailed reports of various kinds such as damages by region, damages by the material used or distribution of damages in the country. There is also the possibility of saving more data for future and use and better documentation of the condition of roads
These reports can then be used to gain meaningful insights into the nature and underlying patterns of damages in the country. The government can then spend resources with more strategic benefit.
This can then greatly help the process of maintaining roads in India and thus by extension, save lives and money.

ACKNOWLEDGEMENT

I take this opportunity to express my sincere gratitude and respect to **CMR Institute of Technology, Bengaluru** for providing me a platform to pursue my studies and carry out my final year project

I have a great pleasure in expressing my deep sense of gratitude to **Dr. Sanjay Jain,** Principal, CMRIT, Bangalore, for his constant encouragement.

I would like to thank **Dr. Prem Kumar Ramesh,** Professor and Head, Department of Computer Science and Engineering, CMRIT, Bangalore, who has been a constant support and encouragement throughout the course of this project.

I consider it a privilege and honor to express my sincere gratitude to my guide
N.Sreedevi, Associate Professor, Department of Computer Science and Engineering, for the valuable guidance throughout the tenure of this review.

I also extend my thanks to all the faculty of Computer Science and Engineering who directly or indirectly encouraged me.

Finally, I would like to thank my parents and friends for all their moral support they have given me during the completion of this work.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1:

INTRODUCTION

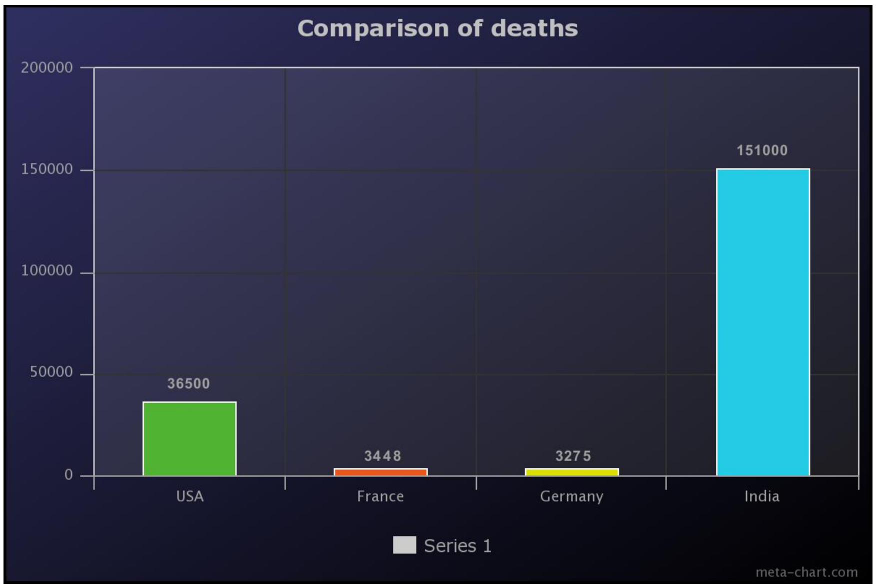

Fig 1.1 Comparison of Deaths per Country

India reports heavy casualties every year due to road traffic accidents.India accounts for only 1% of the world's automobile count however it accounts for 6% of all the traffic deaths.

This is a disproportionate amount which highlights the poor condition of our traffic safety. India has one the largest roadway networks in the world, spanning more than four million kilometres in length. It also boasts to be the busiest.

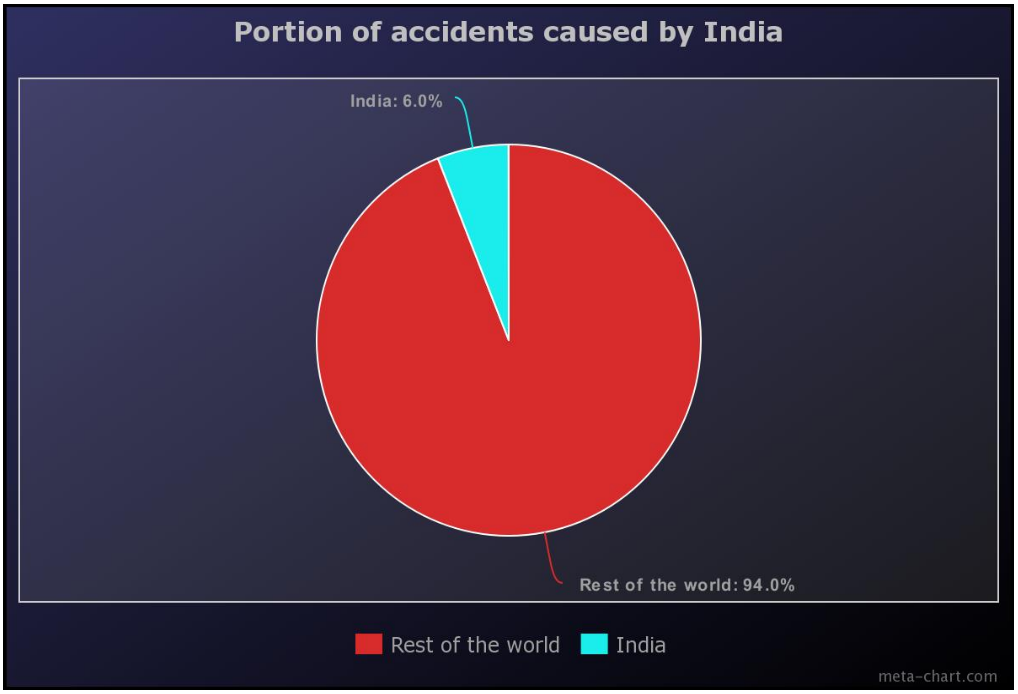

Fig 1.2 Accident Statistics of India vs Rest of World

Unfortunately, India also leads the world in the number of road traffic accidents. Nearly 2,31,000 deaths occur due to road-traffic accidents. A significant portion of which have been linked directly or causally to poorly maintained, worn out roads.

We aim to utilise IoT, Cloud computing and machine learning, three of the most prospective emerging fields of computer-science to help mitigate this problem.

1.1 Relevance of the Project

The ministry of road transport and highways has allocated Rs 3150 crore for the maintenance of the national highways which spans a distance of 1.14 lakh km. This is a small portion of the total 5,897,671 Km of the Indian road network The estimated amount for maintenance of the total road network in India would be 1.62 Lakh crore annually. Such a undertaking is financially impossible leading to a great portion of the roads left poorly maintained

Our project greatly reduces the cost by eliminating the cost of surveying.

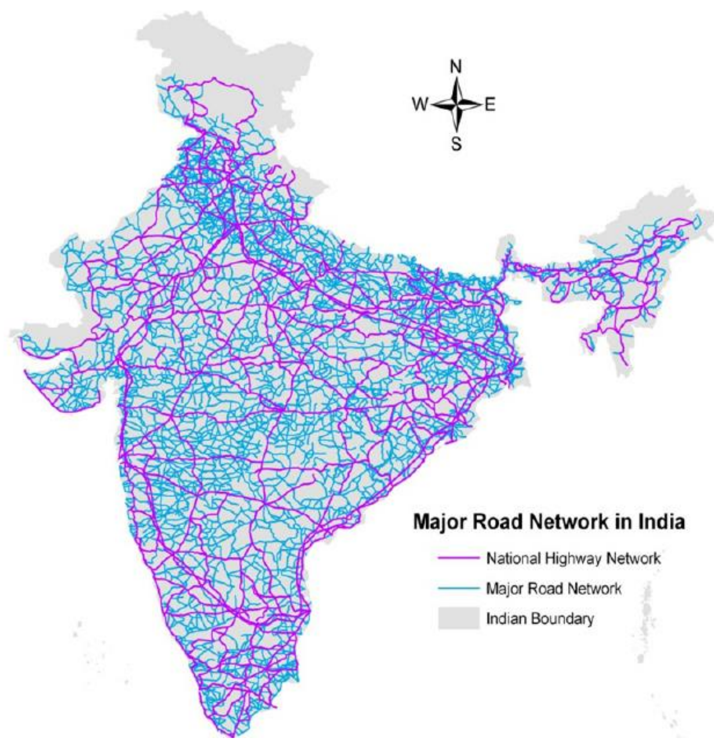

Fig 1.3 Indian Road Network

1.2 Problem Statement

The manual manner in which roads are surveyed is a laborious and costly task. It puts strain on the government both financially and temporally. This leads to lesser surveying of roads. The lack of proper surveying leads to poorly maintained roads with numerous damages and abrasions. These damages cause a lot of casualties in our country.

1.3 Objective

To produce a device that can reduce the human intervention required during the process of road surveying. The device must capably detect damages such as potholes and do so in a cost-effective manner.

The project must also be able to report data in visual manner for easy understanding by humans.

1.4 Scope of the project

The project will aim to provide which can make the surveying and reporting aspects of the maintenance of roads. We aim to demonstrate that road surveying and report generation is an easy task

1.5 Methodology

The aim is to create cost-effective alternative, keeping that in mind it is imperative not to use prohibitively expensive hardware or software frameworks which require great computation power.

At all times the cost of the project and the ease of using it should be prioritised.

CHAPTER 2:

LITERATURE SURVEY

2.1: You Only Look Once (YOLO): Unified, Real-Time Object Detection

we frame object detection as a regression problem to spatially separated bounding boxes and associated class probabilities.

A single neural network predicts bounding boxes and class probabilities directly from full images in one evaluation. Since the whole detection pipeline is a single network, it can be optimized end-to-end directly on detection performance.

Our unified architecture is extremely fast. Our base YOLO model processes images in real-time at 45 frames per second. A smaller version of the network, Fast YOLO, processes an astounding 155 frames per second while still achieving double the mAP of other real-time detectors.

Compared to state-of-the-art detection systems, YOLO makes more localization errors but is less likely to predict false positives on background.

Finally, YOLO learns very general representations of objects. It outperforms other detection methods, including DPM and R-CNN, when generalizing from natural images to other domains like artwork

Advantages:

- Extremely fast
- Simple
- Very High Precision (twice the mean precision as other real time decision systems)
- Global context is accounted for
- Less than half the errors compared to other systems
- Learns global representations (outperformed other detection methods such as DPM and R-CNN by a large margin on art works)

2.1.1 Yolo methodology Unified Detection

Yolo unifies the separate components of object detection into a single neural network. The network uses features from the entire image to predict each bounding box. It also predicts all bounding boxes across all classes for an image simultaneously.

This means the network reasons globally about the full image and all the objects in the image. The YOLO design enables end-to-end training and realtime speeds while maintaining high average precision. Yolo system divides the input image into an S × S grid. If the center of an object falls into a grid cell, that grid cell is responsible for detecting that object. Each grid cell predicts B bounding boxes and confidence scores for those boxes.

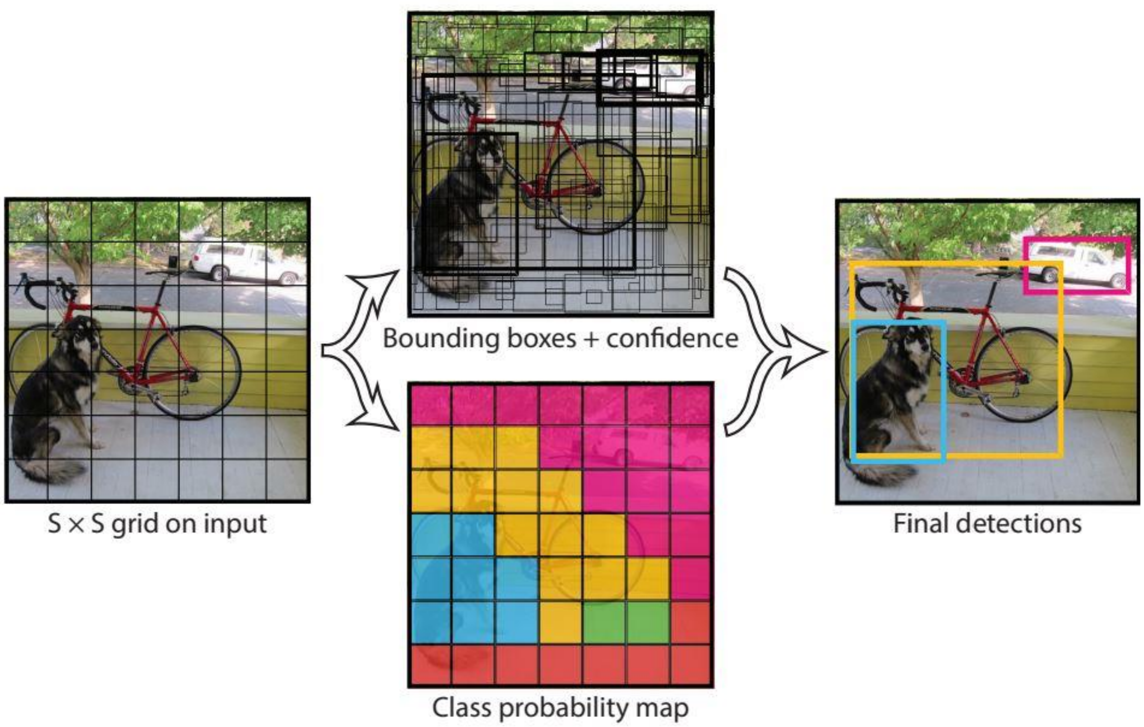

Fig 2.1 YOLO grids

These confidence scores reflect how confident the model is that the box contains an object and also how accurate it thinks the box is that it predicts. Formally we define confidence as Pr(Object) ∗ IOUtruth pred .

If no object exists in that cell, the confidence scores should be zero. Otherwise we want the confidence score to equal the intersection over union (IOU) between the predicted box and the ground truth. Each bounding box consists of 5 predictions: x, y, w, h, and confidence. The (x, y) coordinates represent the center of the box relative to the bounds of the grid cell. The width and height are predicted relative to the whole image.

Finally the confidence prediction represents the IOU between the predicted box and any ground truth box. Each grid cell also predicts C conditional class probabilities, Pr(Classi|Object).

These probabilities are conditioned on the grid cell containing an object. We only predict one set of class probabilities per grid cell, regardless of the number of boxes B. At test time we multiply the conditional class probabilities and the individual box confidence predictions :

Pr(Classi|Object)∗Pr(Object)∗IOUtruth pred = Pr(Classi)∗IOUtruth pred (1) which gives us class-specific confidence scores for each box.

These scores encode both the probability of that class appearing in the box and how well the predicted box fits the object.The yolo designers implement this model as a convolutional neural network and evaluate it on the PASCAL VOC detection dataset.The initial convolutional layers of the network extract features from the image while the fully connected layers predict the output probabilities and coordinates. Our network architecture is inspired by the GoogLeNet model for image classification. The YOLO network has 24 convolutional layers followed by 2 fully connected layers. Instead of the inception modules used by GoogLeNet, we simply use 1 × 1 reduction layers followed by 3 × 3 convolutional layers, similar to Lin et al.They also train a fast version of YOLO designed to push the boundaries of fast object detection. Fast YOLO uses a neural network with fewer convolutional layers (9 instead of 24) and fewer filters in those layers.

2.1.2 Training YOLO

The yolo designers pretrain our convolutional layers on the ImageNet 1000-class competition dataset.
For pretraining we use the first 20 convolutional layers from Figure 3 followed by a average-pooling layer and a fully connected layer. We train this network for approximately a week and achieve a single crop top-5 accuracy of 88% on the ImageNet 2012 validation set, comparable to the GoogLeNet models in Caffe's Model Zoo.
We use the Darknet framework for all training and inference.
–We then convert the model to perform detection. Ren et al. show that adding both convolutional and connected layers to pretrained networks can improve performance.
Following their example, we add four convolutional layers and two fully connected layers with randomly initialized weights. Detection often requires fine-grained visual information so we increase the input resolution of the network from 224 × 224 to 448 × 448. Our final layer predicts both class probabilities and bounding box coordinates. We normalize the bounding box width and height by the image width, height so that they fall between 0 and 1.
The YOLO designers parametrize the bounding box x and y coordinates to be offsets of a particular grid cell location so they are also bounded between 0 and 1. They use a linear activation function for the final layer and all other layers use the following leaky rectified linear activation:

$$\phi(x) = \begin{cases} x, & \text{if } x > 0 \\ 0.1x, & \text{otherwise} \end{cases}$$

They optimize for sum-squared error in the output of our model. They use sum-squared error because it is easy to optimize, however it does not perfectly align with our goal of maximizing average precision.
It weights localization error equally with classification error which may not be ideal. Also, in every image many grid cells do not contain any object. This pushes the "confidence" scores of those cells towards zero, often overpowering the gradient from cells that do contain objects.

This can lead to model instability, causing training to diverge early on. To remedy this, we increase the loss from bounding box coordinate predictions and decrease the loss from confidence predictions for boxes that don't contain objects. They use two parameters, λcoord and λnoobj to accomplish this. We set λcoord = 5 and λnoobj = .5.

Sum-squared error also equally weights errors in large boxes and small boxes. Our error metric should reflect that small deviations in large boxes matter less than in small boxes. To partially address this, they predict the square root of the bounding box width and height instead of the width and height directly.

YOLO predicts multiple bounding boxes per grid cell. At training time, they only want one bounding box predictor to be responsible for each object.

They assign one predictor to be "responsible" for predicting an object based on which prediction has the highest current IOU with the ground truth.

This leads to specialization between the bounding box predictors. Each predictor gets better at predicting certain sizes, aspect ratios, or classes of object, improving overall recall. During training we optimize the following, multi-part. The loss function used by yolo:

$$
\begin{aligned}
&\lambda_{\mathbf{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right] \\
&+ \lambda_{\mathbf{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right] \\
&+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2 \\
&+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2 \\
&+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2 \qquad (3)
\end{aligned}
$$

2.1.3 Limitations of YOLO:

YOLO imposes strong spatial constraints on bounding box predictions since each grid cell only predicts two boxes and can only have one class. This spatial constraint limits the number of nearby objects that our model can predict.

This model struggles with small objects that appear in groups, such as flocks of birds. Since our model learns to predict bounding boxes from data, it struggles to generalize to objects in new or unusual aspect ratios or configurations.

This model also uses relatively coarse features for predicting bounding boxes since our architecture has multiple down-sampling layers from the input image.

Finally, while we train on a loss function that approximates detection performance, our loss function treats errors the same in small bounding boxes versus large bounding boxes.

A small error in a large box is generally benign but a small error in a small box has a much greater effect on IOU. Our main source of error is incorrect localizations.

2.1.4 Inference

Just like in training, predicting detections for a test image only requires one network evaluation. On PASCAL VOC the network predicts 98 bounding boxes per image and class probabilities for each box.

YOLO is extremely fast at test time since it only requires a single network evaluation, unlike classifier-based methods.

The grid design enforces spatial diversity in the bounding box predictions. Often it is clear which grid cell an object falls in to and the network only predicts one box for each object. However, some large objects or objects near the border of multiple cells can be well localized by multiple cells.

Non-maximal suppression can be used to fix these multiple detections. While not critical to performance as it is for R-CNN or DPM, non-maximal suppression adds 2- 3% in mAP

2.2: Other Papers Researched

1. **Xin Jia(2017)** - Research paper aims to review the state-of-the-art in deep learning algorithms in computer vision by highlighting the contributions and challenges from recent research papers. This paper concludes with summarizing the new trends in designing and training deep neural networks, along with several directions.

2. **Simonyan & Zisserman (2015)** - Investigated to know the convolutional network depth on its accuracy in the large-scale image recognition setting. Study concluded convolutional network depth on its accuracy in the large-scale image recognition setting.

3. **Kawano & Yanai (2014)** - In this paper, They reported the feature obtained from the Deep Convolutional Neural Network boosts food recognition accuracy greatly by integrating it with conventional hand-crafted image features. Report concluded that the best classification accuracy, 72.26%, for the UEC-FOOD100 dataset, which proved that that DCNN features can boost classification performance by integrating it with the conventional features.

4. **<u>Hijazi et.al (2014)</u>** - This paper covers the basics of CNNs including a description of the various layers used. It concludes that CNNs give the best performance in pattern/image recognition problems and even outperform humans in certain cases. Cadence has achieved best-in-industry results using proprietary algorithms and architectures with CNNs. We have developed hierarchical CNNs for recognizing traffic signs in the GTSRB, achieving the best performance ever on this dataset. We have developed another algorithm for the performance-versus-complexity tradeoff and have been able to achieve a complexity reduction by a factor of 86 for a CDR degradation of less than 2%. The Tensilica Vision P5 DSP for imaging and computer vision from Cadence has all the features required to implement CNNs in addition to the features required to do image signal processing. More than 850 traffic sign recognitions can be performed running the DSP at 600MHz. The Tensilica Vision P5 DSP from Cadence has an almost ideal set of features to run CNNs.

5. **<u>Zheng et.al (2014)</u>** - This paper proposed a novel part learning approach by a multi-attentional convolution- al neural network (MA-CNN), where part generation and feature learning can reinforce each other. It concludes The proposed network does not need bounding box/part annotations for training and can be trained end-to-end.

CHAPTER 3:

SYSTEM REQUIREMENTS SPECIFICATION

This system is based on machine learning and cloud computing as well as IoT. The requirements include:

3.1 Functional Requirements

- A camera module that can be controlled using a RaspBerryPi proccessing unit
- An optional sensor Unit that can detect the depth/distance of an object (such as sensor)
- A networking module to connect to the internet and connect to the cloud
- A cloud platform to perform the computation
- A Front end GUI to display reports,damage estimations and other similar insights

3.2 Non-Functional Requirements

- The camera module must be able to multiple pictures in a short duration (such a 20 frames per second at the minimum)
- The hardware platform must be inexpensive yet flexible enough to handle the co-ordination of the entire device
- The sensor should have a standard interface which can be accessed by RaspBerryPi
- The cloud platform used must have the option of rapidly provisioning resources such as memory and processing power and release when using it. This is done so the cost of computation in terms of money and the processing needed goes down
- The device on the overall must be small enough for one human to carry without assistance

CHAPTER 4

SYSTEM ANALYSIS AND DESIGN

4.1 Proposed System

We propose an IOT and cloud based solution that can identify potholes and quickly generate reports.

The project will also be capable of giving a damage estimation based on machine learning techniques. The main components of the device and project are shown in figure 4.1.

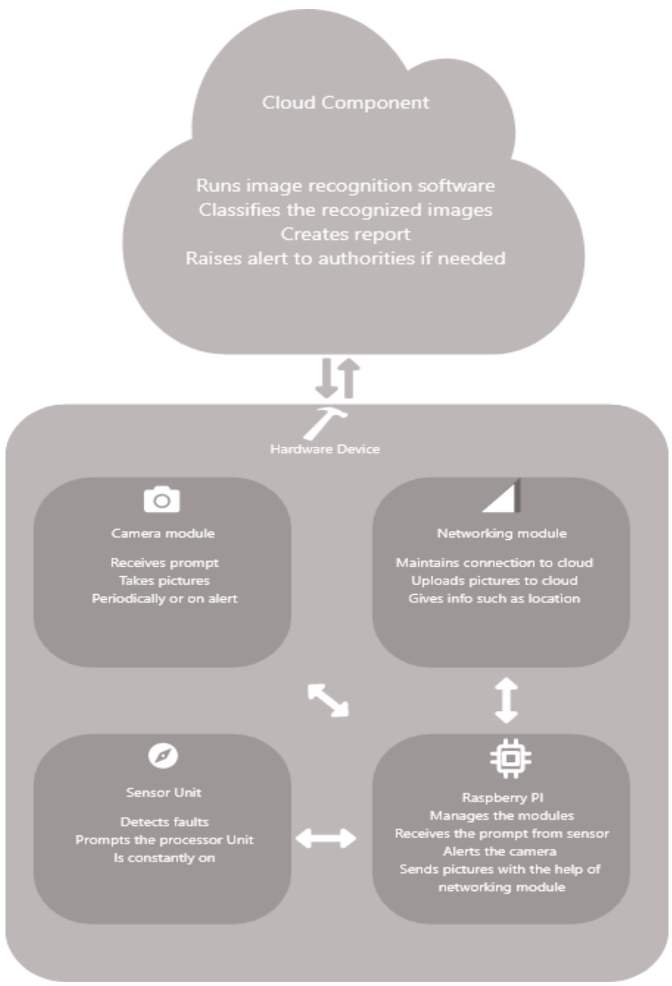

Fig 4.1 Design overview

The hardware device will be equipped with a processing unit (such as Raspberry pi) along with a camera module and a networking module. The device will come equipped with a depth sensor as well. The device will be then fitted over a suitable vehicle which will be traversed along the length of the road to be surveyed. The camera can be configured to take pictures regularly at intervals (Maybe 100 ms) or to take pictures only when the sensor detects an anomaly.

In the first case, the camera takes pictures at intervals and uploads them to the cloud. The cloud component then runs the image recognition software on the pictures to detect damages. If damages are detected then the picture along with its details is saved. Later on these are used to create reports. If the camera is in the second configuration, the sensor comes into play. The sensor (a depth sensor such as SONAR) raises an alert when it detects depth beyond a certain threshold. When the alert is raised the camera takes a picture and uploads it. This consumes lesser electricity and computation power but may be more faulty.

4.2 Development Tools

4.2.1 Hardware

- The main hardware is a Raspberry Pi for control operations.
- It receives alerts from the sensor and alerts the camera, it invokes the networking module to upload the pictures to the cloud.
- We also use a camera module which takes pictures.
- The networking module will connect the hardware device to the cloud where the actual image processing will take place.
- The sensor unit will always be on the lookout for damages and if found will raise an alert.

4.2.2 Software

❖ **MongoDB** : MongoDB is a cross-platform document-oriented database program. Classified as a NoSQL database program, MongoDB uses JSON-like documents with optional schemas. MongoDB is developed by MongoDB Inc. and licensed under the Server Side Public License.

❖ **Javascript** : JavaScript, often abbreviated as JS, is a programming language that conforms to the ECMAScript specification. JavaScript is high-level, often just-in-time compiled, and multi-paradigm. It has curly-bracket syntax, dynamic typing, prototype-based object-orientation, and first-class functions.

- **Node.js** : Node.js is an open-source, cross-platform, JavaScript runtime environment that executes JavaScript code outside a web browser. Node.js lets developers use JavaScript to write command line tools and for server-side scripting—running scripts server-side to produce dynamic web page content before the page is sent to the user's web browser. Consequently, Node.js represents a "JavaScript everywhere" paradigm, Unifying web-application development around a single programming language, rather than different languages for server- and client-side scripts.

- **React.js** : React is an open-source JavaScript library for building user interfaces. It is maintained by Facebook and a community of individual developers and companies. React can be used as a base in the development of single-page or mobile applications.

- **Express.js** : Express.js, or simply Express, is a web application framework for Node.js, released as free and open-source software under the MIT License. It is designed for building web applications and APIs. It has been called the de facto standard server framework for Node.js.

- **Google Cloud Platform SDK** : Google Cloud Platform, offered by Google, is a suite of cloud computing services that runs on the same infrastructure that Google uses internally for its end-user products, such as Google Search, Gmail and YouTube.

- **YOLO v3 for training** : YOLOv3 is extremely fast and accurate. In mAP measured at .5 IOU YOLOv3 is on par with Focal Loss but about 4x faster. Moreover, you can easily trade off between speed and accuracy simply by changing the size of the model, no retraining required.

- **Keras** : Keras is an open-source neural-network library written in Python. It is capable of running on top of TensorFlow, Microsoft Cognitive Toolkit, R, Theano, or PlaidML. Designed to enable fast experimentation with deep neural networks, it focuses on being user-friendly, modular, and extensible.

- **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.

4.3 Dataset

We need a database of pictures taken of potholes from a variety of roads in various angles. There are multiple datasets available. For the purpose of this project to demonstrate it's working we have used a dataset found in kaggle.
The dataset has well over 688 images. Though it will suffice for this project's demonstration to train the model over these images, it may be useful to find more images to train the model on.

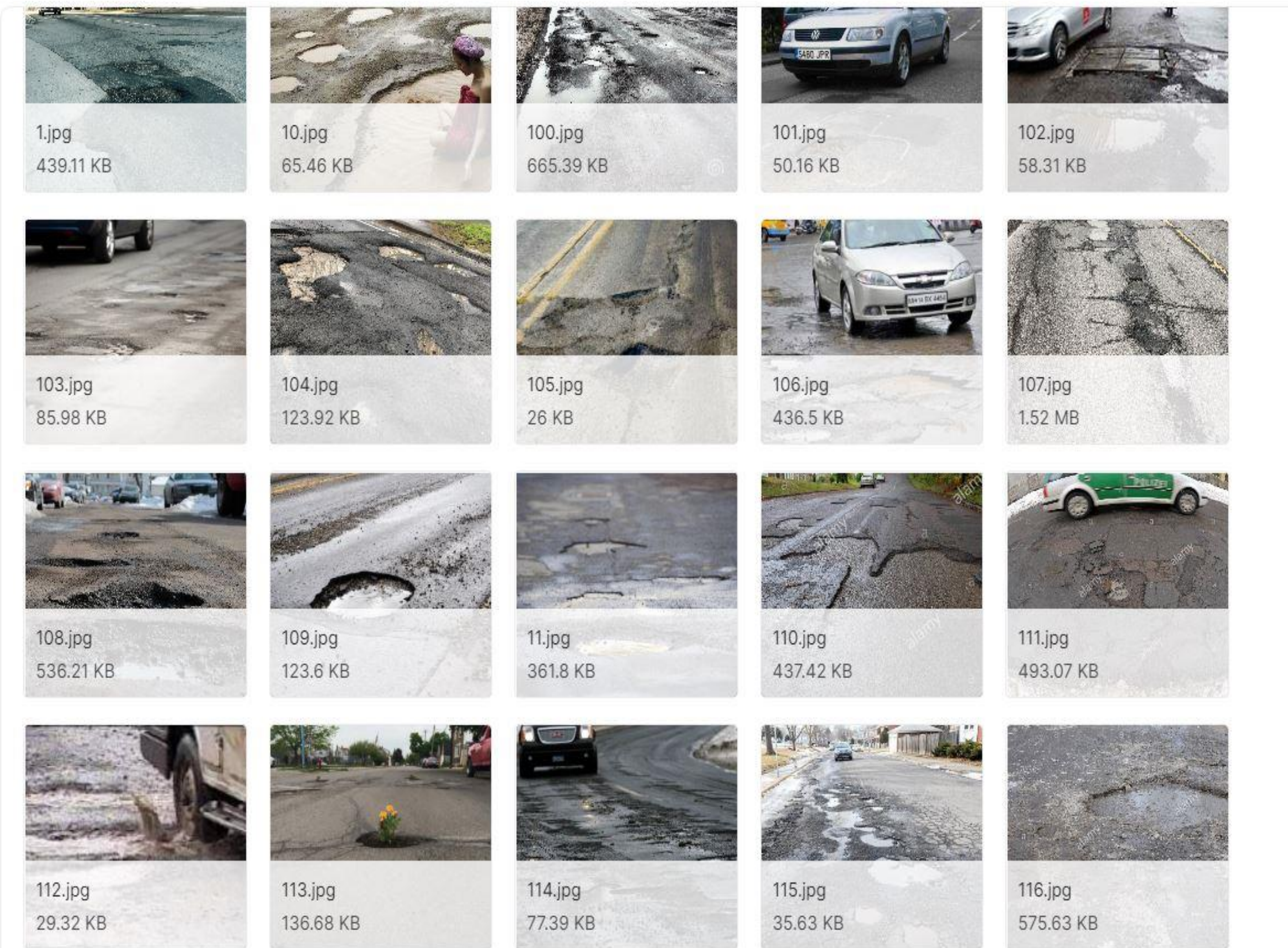

Fig 4.2 Dataset of potholes

4.4 Flowchart of operation

The general flow of work is shown in figure 4.3.
The hardware device can come in two configurations, one with a sensor and the other without a sensor

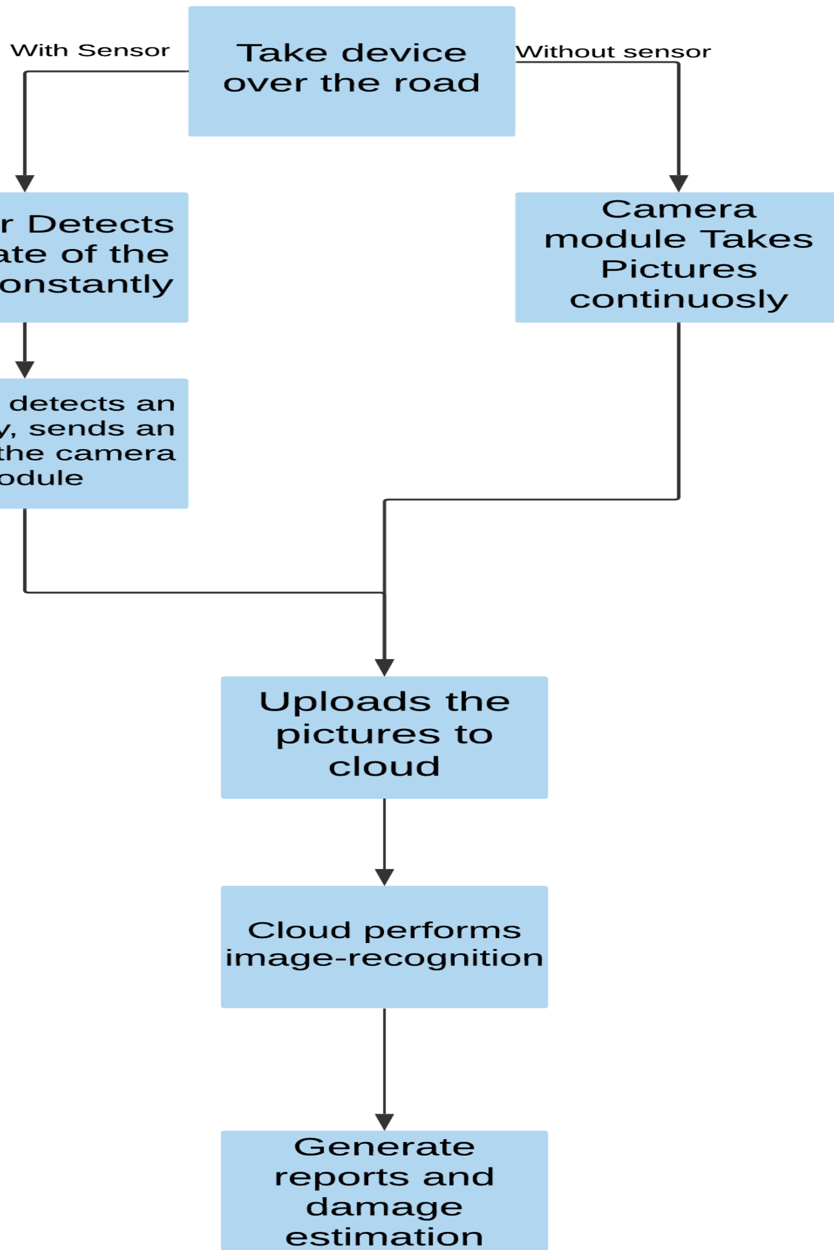

Fig 4.3 workflow

CHAPTER 5

IMPLEMENTATION

We Implemented the solution as an IOT device, The data from the device is processed on the virtual machine instance on a cloud platform like GCP / AWS. Data processing results are then stored in a No-SQL Managed database like DynamoDB or FireStore, Final step involves creating API endpoints to fetch the result data and display them as a web application using React.js

5.1 IOT Setup

IOT Setup consists of a Raspberry Pi with the camera module mounted on it, the Raspberry Pi is mounted on a low flight drone.Raspberry Pi is connected to the Google Cloud VM instance via the wifi connection. Google cloud SDK is utilized on the raspberry pi end to authenticate with the GCP server instance and send data Images in the realtime.

The data once transmitted from the device to the cloud in real time the processing on the data happens. Results are stored in a No-SQL managed database using REST API endpoints. API’s are made available to fetch the results from the database.

5.2 Working Model

We aim to create a device that can be taken over the length of the road, the device comes fitted with a camera module that can be run in two modes to take pictures of the road.

In the first mode, the camera module takes pictures at regular intervals (100 ms) and uploads them to cloud continuously.

In the second mode, a sensor mounted on to the device (such as sonar) detects irregularities in depth and alerts the camera.

Upon raising the alert, the camera takes a snapshot of the picture and uploads it to the cloud.

Within the cloud, the image recognition software (we have chosen YOLO) runs. If a pothole or other damage is detected then it is recorded and an estimate is generated based on the depth and number of potholes and their size(s).

The estimation is done using machine learning techniques, specifically clustering and random forest regression.

Fig 5.1 Working model abstract

Both the modes have their ups and downs, in the first mode, damage detection is better but there is a significant power and computation required. The added benefit is the lower cost due to the absence of the sensor module.

The second mode is far more computationally sensible however the results are worse off.

Fig 5.2 Components overview

CHAPTER 6

RESULTS AND DISCUSSION

The inner workings of the device and project are abstracted from the end user. An end user maybe an immediate supervisor in charge of the surveying of roads or a member of the planning committee who intends to make a detailed report and plans for the future.The end user will see a dashboard similar to the one shown below (varies based on the kind of device used).

6.1 Dashboard

The front end will be web GUI as shown below. The dashboard will be a place where the various insights at a glance

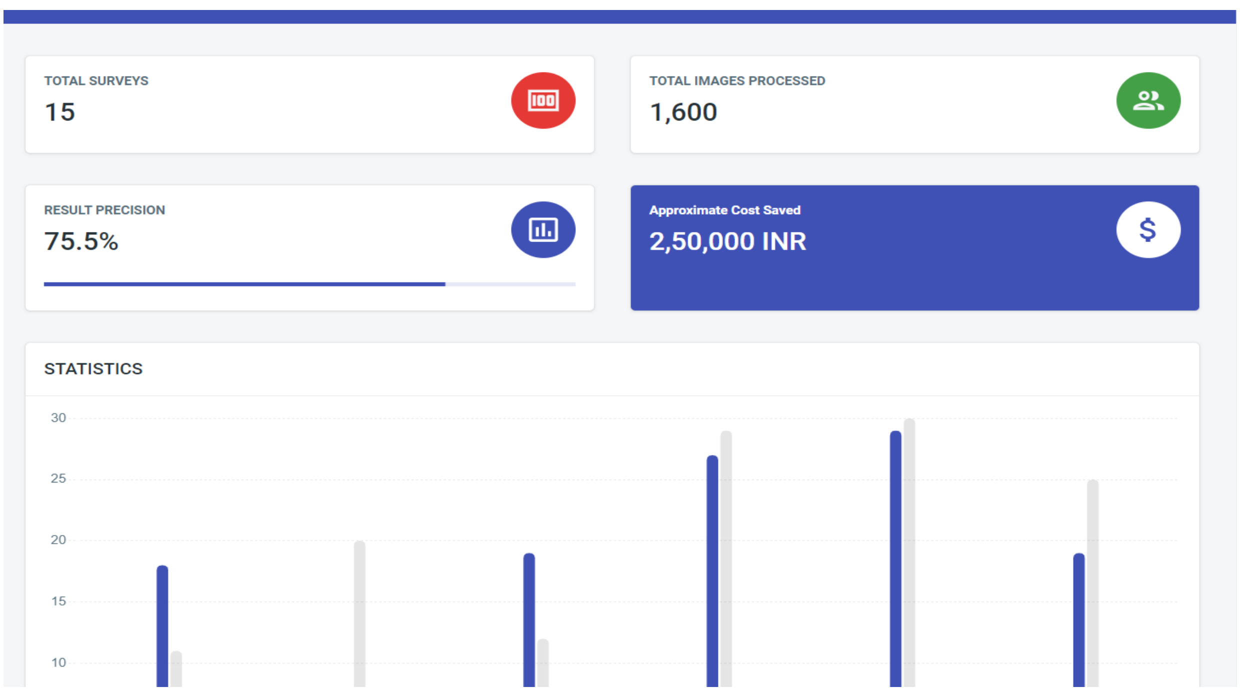

Fig 6.1 Dashboard Screenshot

6.2 Reports

The dashboard will come featuring the option wo select various kinds of reports and charts such as bar charts, point graphs, line graphs, pie charts, dial charts and so on.

The below figure shows one kind of chart, the pie-chart (albeit with an aesthetic design).

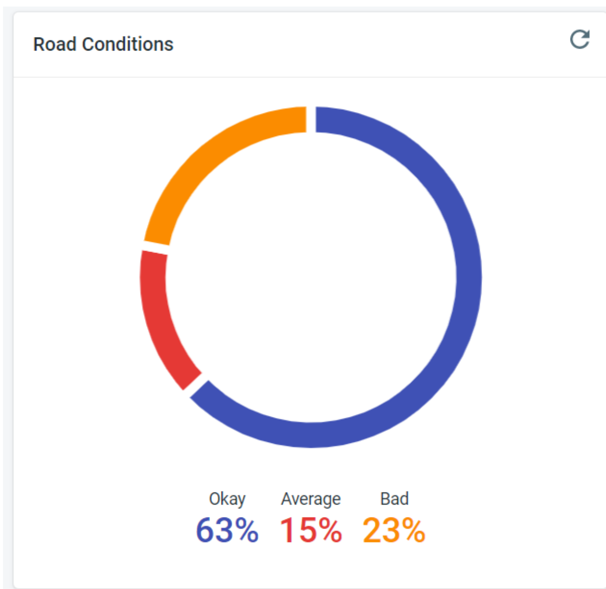

Fig 6.2 Report template

6.3 Damage Estimation

We propose a system that can greatly reduce the monetary and labour costs of surveying roads. This can reduce the strain on the government during the maintenance of the roads. It can give intelligent insights such as estimation and which region has most road damages and so on.

CHAPTER 7

TESTING

Large dataset of pothole pictures and highway were downloaded from kaggle. The images imitated the images that were to be taken by the raspberry pi camera module.

The images were run against the pothole detection program on a cloud machine environment, Results were observed and automatically stored in a database,

The API endpoints that were created gave the real time information as they data was being processed and analysed by the cloud virtual machine.

7.1 Accuracy Demonstration

The various images we tested the model on worked satisfactorily. Figure 7.1 shows the recognition of multiple potholes in the same picture with good accuracy

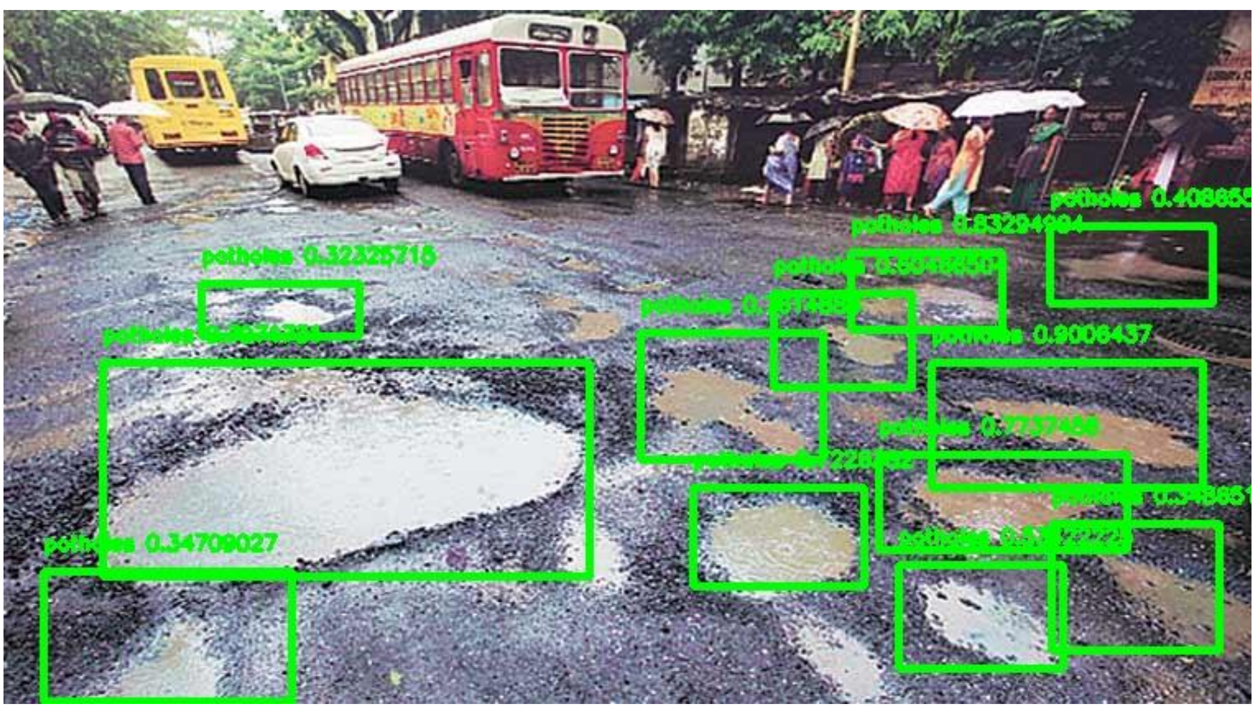

Fig 7.1 Tested pothole image

CHAPTER 8

CONCLUSION AND FUTURE SCOPE

8.1 Conclusion

The project successfully demonstrated that the process of surveying is possible without the crippling financial burden nor the intensive labour required by the current methodology of surveying of roads in India.

8.2 Contribution

We made an automated system that reduces human intervention to a minimum in the process of surveying roads.
We also made the system capable of offering deep, meaningful insights into the data and reports generated.

8.3 Future Scope

The project has the scope of being implemented on a national scale. The entire process of road surveying as the potential of being overhauled.
It can save great amounts of money and labour. The freed up resources can now be focused on other work that was previously left unattended to.

REFERENCES

- Road traffic accidents in india 2018: https://scroll.in/article/944201
- Road traffic accidents in UK,:https://www.gov.uk/government/statistics/reported-road-casualties-in-great-britain-provisional-estimates-year-ending-june-2018
- Road accidents in usa: https://www.nhtsa.gov/traffic-deaths-2018
- Indian expenditure on road maintenance as given by the following website: https://www.prsindia.org/parliamenttrack/budgets/demand-grants-analysis-road-transport-and-highways
- Road statistics India: https://en.wikipedia.org/wiki/Roads_in_India
- Jia.X (2017); Image Recognition Method Based on Deep Learning, *Tianjin University of Technology*.
- Simonyan.K & Zisserman.A (2015); VERY DEEP CONVOLUTIONAL NETWORKS FOR LARGE-SCALE IMAGE RECOGNITION, *University of Oxford.*
- Kawano.Y & Yanai.K (2014); Food Image Recognition with Deep Convolutional Features, *Department of Informatics The University of Electro-Communications, Tokyo.*
- Hijazi et.al (2014); Using Convolutional Neural Networks for Image Recognition, *Cadence*
- Zheng.h et.al (2014); Learning Multi-Attention Convolutional Neural Network for Fine-Grained Image Recognition, *1University of Science and Technology of China, Hefei, China*