International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 08 Issue: 05 | May 2021 www.irjet.net p-ISSN: 2395-0072

Spothole: A Real Time Object Detection Application using Modern Deep Learning Technique

Divya Jain[1], Ankita Gavali[2], Sameer Kamble[3], Dr. Amol Pande[4]

[1]*Student, Dept. of Computer Engineering, Datta Meghe College Of Engineering, Navi Mumbai, India*
[2]*Student, Dept. of Computer Engineering, Datta Meghe College Of Engineering, Navi Mumbai, India*
[3]*Student, Dept. of Computer Engineering, Datta Meghe College Of Engineering, Navi Mumbai, India*
[4]*H.O.D, Dept. of Computer Engineering, Datta Meghe College Of Engineering, Navi Mumbai, India*

Abstract - *As low quality of construction material is being used in India road accidents are a very common problem also its detection and avoidance is very difficult and challenging task. Due to the above problems, roads get damaged early and potholes appear on the roads which cause accidents. Year after year, the accident rates are increasing due to the up-surging potholes count. In most of the places road maintenance process is done manually hence it takes long time and human labor to complete the task of detection. Thus, there is a growing need for a cost-effective automated identification of potholes. In this paper we propose an android application for detecting potholes in real time. Using Transfer Learning we have trained Deep Learning based Single Shot Detector (SSD) model for object detection. Results show that our model performs optimally for all shapes of pothole along with good accuracy.*

Key Words: **Pothole Detection, Deep Learning, Object Detection, Transfer Learning, SSD**

1. INTRODUCTION

India has one of the largest road network in the world, spanning 5.89 million kilometers (kms) overall.. Road transportation has slowly increased over the past years with improvement in connectivity between cities, towns and villages in the country [1].Road maintenance is a challenging task. Potholes are a very common thing in a country like India. Injuries and deaths happening due to road accidents are taken very lightly therefore every year around 3600 people die due to potholes. More than 30% of death has taken place due to potholes. The Ministry of Road Transport and Highways provided figures that over 9300 deaths, 25000 injured in the last three years due to potholes and more than 25,000 people are getting injured due to potholes. But still, it is not taken as a serious issue in India. Still, potholes are getting originated [2]. Municipal Authority of the city, there are irregularities in inspecting the road and the potholes so the user as being the citizen can help in complaining and show the urgency that require immediate action. The paper shows the work on two sources of database one being an open dataset available in Kaggle [3] and second source of images of current state of roads. An object detection model is trained to detect the potholes using the dataset. The focus is also on creating an application which is user friendly where in user can register the pothole complaint. The project's main aim is to provide a solution to automate the manual challenges of finding the potholes by the municipal body. The system can detect the real time capture image along with the latitude and longitude pothole area and then the user complain is register which will be mounted on real time database and is accessible to the authority. This information can be used by the Municipal Authority to plan their action on the reparation work of potholes.

The paper is organized as follows: Section II contains the literature survey for the project. Section III explains the Methodology. Section IV explains Data Pre-Processing. Section V shows Experimental Results. Section VI concludes the paper.

2. LITERATURE REVIEW

MIT *World Peace University, Pune, India* have proposed "Deep Learning Approach to Detect Potholes in Real-Time using Smartphone" [4]. In this paper, The system follows two-fold cross verification mechanism to detect potholes using the cameraas well as accelerometer and gyroscope sensors of the Android device. Amita Dhiman and Reinhard Klette proposed "Pothole Detection Using Computer Vision and Learning" [5]. In this research, four different techniques are proposed and tested against each other. Each technique has its own benefits and can provide different pathways to a number of applications. The LM1 model can identify a pothole under challenging weather conditions with good precision and recall whereas the LM2 model is capable of real-time pothole identification. The SV2 approach can identify potholes and road manifolds with very high accuracy when used with stereo-vision cameras.

Pereira, V. et. al. in [6] have used Convolution Neural Network and compared the performance of their model with SVM and found that their model out formed SVM with 99.80% accuracy. They deployed the model using CNN,

pooling, ReLU activation function, Adam Optimizer, and Sigmoid function. Ping Ping, Xiaohui Yang and Zeyu Gao proposed "A Deep Learning Approach for Street Pothole Detection" [7].This paper proposes an efficient pothole detection system using deep learning algorithms which can detect potholes on the road with only a camera attached to the dash of a car and an internet connection. Artis Mednis proposed "Real Time Pothole Detection using Android Smartphones with Accelerometers" [8]. Thispaper describes accelerometer data based pothole detection algorithms for deployment on devices with limited hardware/software resources and their evaluation on real world data acquired using different Android OS based smart- phones. Lin, J. et. al. in [9] have used a nonlinear SVM model classification tool with a Gaussian radial basis function for thedetection of a pothole.

3. METHEDOLOGY

3.1 TensorFlow Object Detection API:

The TensorFlow object detection API is the framework designed for solving problems such as object detection, Image classification etc. using deep neural networks.. The framework consists of pre trained neural networks which are located in TensorFlow Model Zoo. This includes a set of pretrained models trained on the COCO dataset, the KITTI dataset, and the Open Images Dataset. Each model has its own mAP value as well speed of detection. Various models are used for various purpose and we can make use of our own dataset to train them for our desired work.

3.2 Single Shot Detector(SSD) Object Detection Model:

It is one of the pre trained model used in object detection as well as image classification. Objects are detected in a Single Shot by making use of multibox in Single Shot Detector model. It is ominously faster in speed and high accuracy object detection algorithm.
Reason behind SSD having high speed and accuracy is because it:

1. Eliminates bounding box proposals like the ones used inRCNN's
2. Includes a progressively decreasing convolutional filter for predicting object categories and offsets in bounding box locations.

Architecture of Single Shot Detector:
SSD has a base VGG-16 network followed by multibox convolution layers
Base neural network: Main task is feature extraction
VGG-16 base network for SDD is standard CNN architecture for high quality image classification which doesn't consists of the final classification layers. VGG-16 is used for feature extraction.
Additional Conv Layers: Main task is detection of objects
Additional convolutional layers are added to base VGG network for detecting objects. At the end of the base network the size of convolutional layers decreases progressively which helps in detecting objects at multiple scales. The convolutional model for detection is different for each feature layer.

By using multiple features of different sizes having multiple scales prediction for the bounding boxes and confidence score for objects present in the image is done. As convolutional layers size decreases progressively it decreases feature map size and increases depth. The deep layers cover larger receptive fields and construct more intangible representations which helps in detecting larger objects. Smaller objects in the image are detected by using initial conv layers.

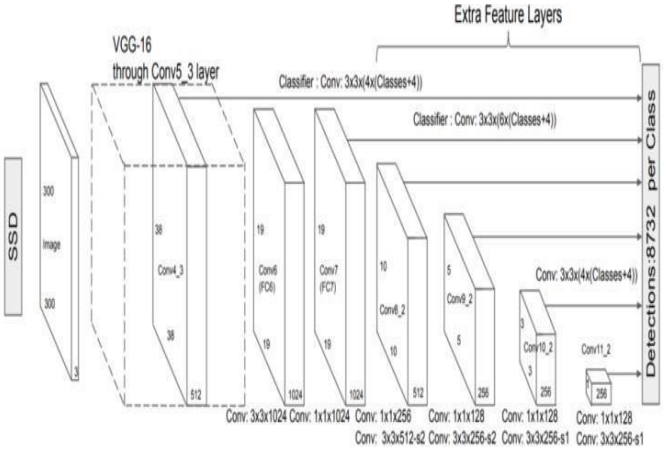

Fig1: SSD model Architecture

An image having ground truth bounding boxes for every object in the image is passed as an input to SSD model. As mentioned above VGG-16 base neural network extracts the features from the imageand then conv layers takes in boxes of different aspect ratio with different scales and checks for objects at each location in the image. Multiple boxes of different sizes, aspect ratio can be seen across the entire image. SSD uses 8732 boxes. The box that mostoverlaps with the ground truth bounding box containing objects isselected. The final output is an image along with that bounding box surrounding the object which was selected on the basis of ground truth. One thing to note is that IoU (intersection over union) ratio should be greater than 0.5 between predicted boxes abd ground truth.

3.2.1 ResNet-50 Architecture:

ResNet which stands for Residual Networks is one of the widely used model in computer vision tasks. This model won the ImageNetchallenge back in 2015.This model allows us to

train extreme deep neural networks with 150+layers

effectively hence it is so popular. Before ResNet, training of extreme deep neural networks was difficult because of vanishing gradients problem. The ResNet- 50 model has 5 stages each having a convolutional and identity block . Each convolutional block has 3 convolution layers and each identity block also has 3 convolution layers. The ResNet-50 consists of more than 23 million trainable parameters.

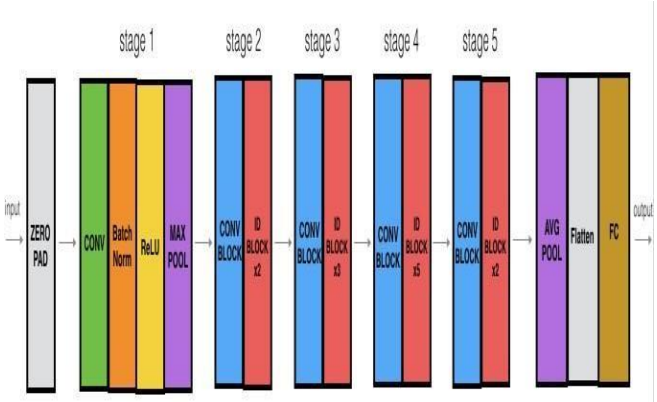

Fig2: ResNet-Architecture

For achieving higher accuracy Deep Learning community started building deeper neural networks. Building deep neural networks have strengthen the performance of the models as they are able to represent more complex features. But everything comes up with some disadvantage in this case training up more layers resulted in accuracy degradation. The main reason for accuracy decreasing abruptly was vanishing gradient problem which is clearly observed in deeper networks.

The solution for this problem was addressed by the authors by introducing deep residual learning framework which provides shortcut connections for performing identity mappings.

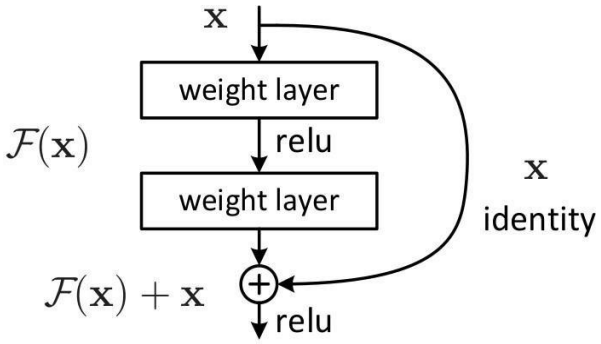

Fig3: ResNet-building block

They explicitly let the layers fit a residual mapping and denoted that as H(x) and they let the nonlinear layers fit another mapping F(x):=H(x)−x so the original mapping becomes H(x):=F(x)+x as can be seen in Fig---.And the benefit of these shortcut identity mapping were that there was no additional parameters added to the model and also the computational time was kept in check.

3.2.2 MobileNet-V2:

MobileNet models are used for many tasks such as detection, classification, segmentation etc. because they are small, have low-latency and low-power which provides high speed in detection purposes. One of the biggest advantage or pluspoint of MobileNet models are their compatibility with android that is they run on android devices very efficiently when integrated using TensorFlow Lite.

As you can see in fig4 MobileNet have two types of block, one is residual block having stride equal to 1 and another is for downsizing with stride equal to 2.There are threelayers in each block, first one is the Conv 1×1 layer having Relu6,second is depth wise 3×3 convolution layer which applies a single filter to each input channel. Third one is non linear 1×1 Conv layer.

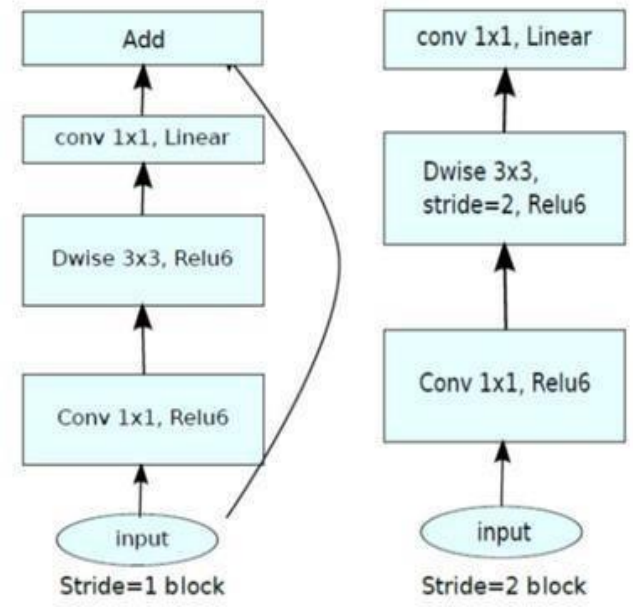

Fig4: MobileNet Structure.

MobileNetV2 architecture is made up of an inverted residual structure such that input and output of the residual block consists of thin bottleneck layers which are unlike the older models which used expanded representation in the input. As mentioned above there is a depth wise convolution layer in MobileNet V2 models which filters outs features in intermediary expansion stage that helps to remove non linearities in the below narrow layers in order to maintain representational power.

3.3 Transfer Learning:

Transfer learning is one of the most popular concept in deep learning which is widely used in solving real world problems. Transfer learning means using pre trained model on variety of new problems by making use of our own label set. Basically we have our own dataset along with labels(classes

We want to classify or detect via our model)and we take a pre trained model(model trained on coco dataset) which already has knowledge of some problems on which it was trained and now we transfer that knowledge to our problem so that we get our desired output. To cut a long story short, we make use of knowledge already gained by the model on some problem and apply it on some related problem. . With transfer learning, we try to achieve what has been learned in one task to improve generalization in another. We transfer the weights that a networkhas gained at "task X" to a new "task Y.

"The basic idea behind using transfer learning is taking advantage of the knowledge a model has learned from a task with a lot of available labeled training data and transferring it to a new task that doesn't have much data. Instead of starting the learning process from beginning, we go with patterns learned from solving a related task.

3.4 Android Firebase:

Firebase is a platform for developing mobile application provided by Google. It provides variety of tools and services to make quality apps. It is a NoSQL database program which stores data in JSON kind of format. It is highly robust as it allows user to restart its uploads and downloads from where he left thereby saving users time and bandwidth. It is highly scalable as it provides huge storage space and with strong security. Main advantage of using Firebase, it is built in service provided in Android Studio hence connecting it with our application was easier with no overhead of any external API or third party service.

3.4.1 Real-Time Database:

The Firebase Realtime Database[10] it is NoSQL database and ishosted on cloud. It helps to sync the data and store it in real time. It also work in offline mode because of Real Time Database SDKthat uses the local cache of devices and store the changes. Due to which when the device in online local data is synchronized automatically. The real time syncing is very useful and easy on any device for accessing the data.

3.4.2 Cloud Storage:

Cloud storage[11] it is used for storing of photos and videos. It is very easy and efficient to store the content which is user generated. The main aim of cloud storage is to automatically handle the offline and online transfers and help the app to regain the connectivity for this the firebase SDK is used. This helps the user in saving time and gives efficiency.

3.5 Tensorflow Lite :

To run tensorflow models on various devices like mobile, embedded and IoT we need to add tflite file to respective devices, for generating this tflite file and performing various tflite operations we have a framework called Tensorflow Lite. It enables on-device machine learning inference with low latency and a small binary size. TensorFlow Lite consists of two main components: first is the TensorFlow Lite interpreter, which is specially design to keep mobile apps smaller and faster, second is TensorFlow Lite converter, which converts TensorFlow models into FlatBuffer format which reduces the model size and via this FlatBuffers we could perform Tensorflow Lite operations.

4. DATA PRE-PROCESSING

We collected 614 pothole images to create our dataset. After creating our dataset we resized all the images to 320 * 320 dimension because our training model expects a input image of that dimension. After resizing we labelled all images, labelling means marking bounding boxes around the object/class we want to detect, in our case the class name was pothole. After labelling, xml file for each image is created which is useful for annotation purpose that is to generate CSV file and TfRecord file which are necessary files prior starting the training of the model. Our dataset was divided into training data which consisted of 550 images and test data which consisted of 64 images.

5. EXPERIMENT - TRAINING AND INTEGRATION :

Once the TFRecords files are generated, next step is fine tuning parameters in the config file such as number of detection, batch size, training steps, iou ratio, etc. Once the config file is ready next step is to put model on training, as soon as training starts we can see the loss value of the model after the training process we need to save the model so that we can export it for transfer learning i.e we can generate the tflite model file out of it, this tflite file is necessary because to integrate model with android app this file is required along with label map file(a text file containing the name of the classes we want to detect).Below are the experimental comparison results of the two model on which we performed the training, fig 5.1, 5.2 and fig 6.1, 6.2 shows single pothole results while fig 7.1 , 7.2 shows multi pothole evaluation, figures in the left side shows test image accuracy of SSD-ResNet model while figures in the right shows test image accuracy of SSD-MobileNet model.

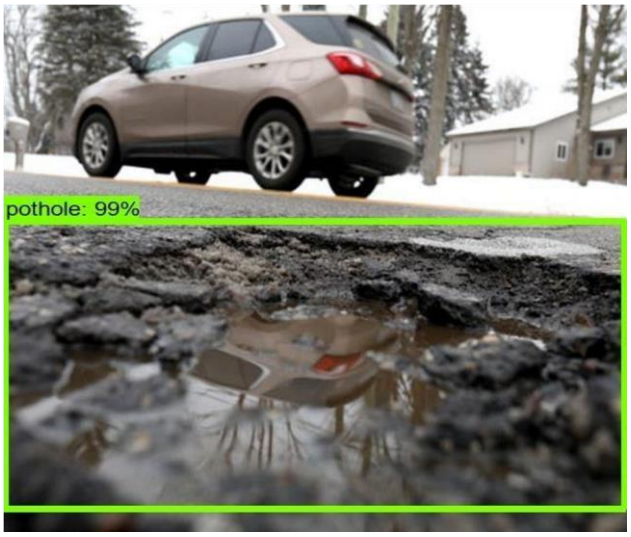

Fig5.1 : ResNet Test Image 1

Fig5.2: MobileNet Test Image 1

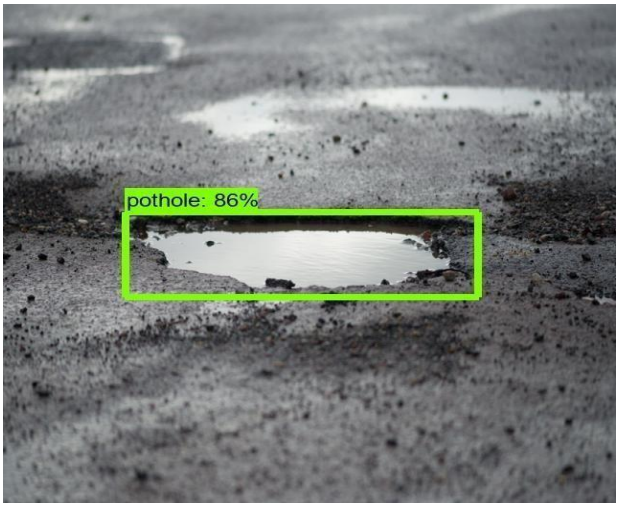

Fig6.1 : ResNet Test Image 2

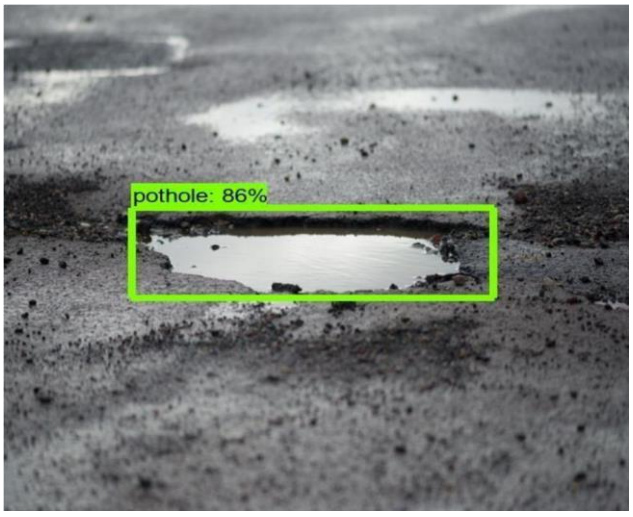

Fig6.2 : MobileNet Test Image 2

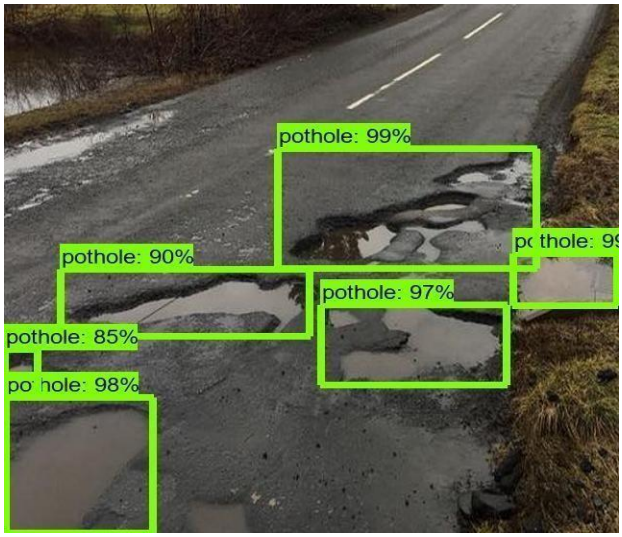

Fig7.1 : ResNet Test Image 3

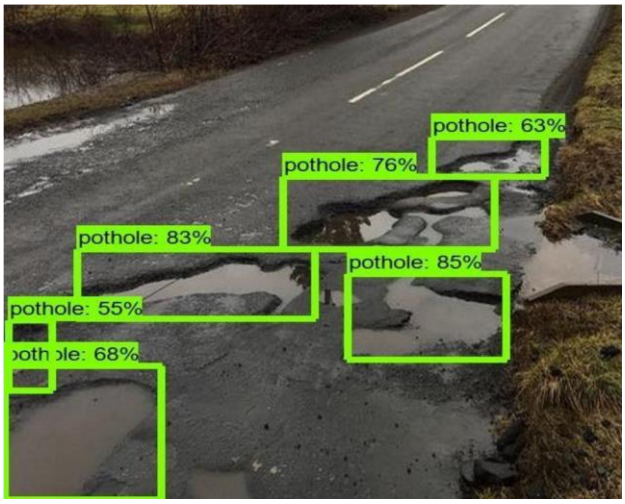

Fig7.2 :MobileNet Test Image 3

6. RESULTS

SSD-ResNet model was trained for 25000 steps taking a batch size of 4 and SSD-MobileNet model was trained for 50000 steps takingthe same size.

6.1 SSD-Resnet:

Fig 8 describes average precision and recall values for SSD ResNet model for varying IoU ratio from 0.50 to 0.95.Interesection Over Union(IoU) is simple ratio that denotes the match in area for the bounding boxes we created across the object when labelling the data and the model predicted bounding box. IoU greater than 0.5 is considered to be hit,else if it is less than 0.5 than your model fails predicting. Here in the Fig 8 SSD Resnet model's evaluated average precision and recall are shown for IoU ratio between 0.50 and 0.95.Second observation to be noted in Fig 8 is precision and recall value for different dimensions of object. You can see in the Fig8 values of area small, area medium and area large objects, what this indicates is, if area of the object in the image < 32 * 32 px than that object is considered small, if area is between 32 * 32 px and 96 * 96 px ,i.e. 32*32 < area < 96 * 96 then the object is medium in size and if area > 96 * 96 px then it is considered as large size object. For small size average precision as well as average recall value is -1 that is this model didn't detected smaller size object at all in test images(of the reason is that our test dataset was small and mostly consisted of larger objects).Average precision and recall value for medium size objects is equal to 0.240 and 0.540 respectively, and that for largesize objects is 0.467 and 0.595 respectively.

```
Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.423
Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.734
Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.453
Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = -1.000
Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.240
Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.467
Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.306
Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.533
Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.586
Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = -1.000
Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.540
Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.595
```

Fig8: Average Precision and Recall values for all size of objects.

In Fig 9 we have shown the evaluation of various losses. First is the localization loss which means how well model has placed or predicted the offset of the bounding boxes on the object, the valuefor this loss in SSD ResNet is 0.218.Second is the classification loss which states how well the model classifies the required class(i.e potholes) in the image, the value for this loss is 0.688.Third is the regularization loss which describes how well is the model optimized, the value for this loss is 0.199.Total Loss is calculated as sum of all the above losses and the value is 1.02 forSSD ResNet model

```
INFO:tensorflow:       + Loss/localization_loss: 0.218348
I0424 05:20:04.003155 140087953090432 model_lib_v2.py:979]     + Loss/localization_loss: 0.218348
INFO:tensorflow:       + Loss/classification_loss: 0.688197
I0424 05:20:04.004103 140087953090432 model_lib_v2.py:979]     + Loss/classification_loss: 0.688197
INFO:tensorflow:       + Loss/regularization_loss: 0.199326
I0424 05:20:04.005017 140087953090432 model_lib_v2.py:979]     + Loss/regularization_loss: 0.199326
INFO:tensorflow:       + Loss/total_loss: 1.025871
I0424 05:20:04.005970 140087953090432 model_lib_v2.py:979]     + Loss/total_loss: 1.025871
```

Fig9: Loss values

6.2 SSD-MobileNet:

Fig 10 shows the IoU ratio along with average precision and recall values for MobileNet model. As you can see in Fig 10 model is evaluated for IoU ratio varying from 0.50 to 0.95.The average precision and recall value for smaller size object in MobileNet model is 0.202 and 0.200 respectively. Average precision and recall value for medium size object is 0.395 and 0.605 respectively ,and that for large size object is 0.664 and 0.755 respectively.Fig 11 shows loss value for different kind of losses. The localization loss value for this model is 0.114, classification loss value is 0.316 and regularization loss is 0.088.Total loss value is0.519.

As you can see the clear difference in Average precision and recall values as well as Loss values for SSD-ResNet and MobileNet model. MobileNet detects smaller objects which wereundetected in SSD ResNet, also you can see it has higher precision in detecting medium and large size objects. Lesser the value better is model for detection, clearly observed in results loss value of MobileNet is much better than ResNet. Due to above results and the biggest advantage of deployment on android we choose MobileNet model for our application.

```
Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.554
Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.838
Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.635
Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.202
Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.395
Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.664
Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.443
Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.658
Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.680
Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.200
Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.605
Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.755
```

Fig10 : Average Precision and Recall values for all size of objects

```
INFO:tensorflow:       + Loss/localization_loss: 0.114797
I0430 05:26:45.987304 139760939685760 model_lib_v2.py:979]     + Loss/localization_loss: 0.114797
INFO:tensorflow:       + Loss/classification_loss: 0.316287
I0430 05:26:45.988516 139760939685760 model_lib_v2.py:979]     + Loss/classification_loss: 0.316287
INFO:tensorflow:       + Loss/regularization_loss: 0.088692
I0430 05:26:45.989844 139760939685760 model_lib_v2.py:979]     + Loss/regularization_loss: 0.088692
INFO:tensorflow:       + Loss/total_loss: 0.519776
I0430 05:26:45.991113 139760939685760 model_lib_v2.py:979]     + Loss/total_loss: 0.519776
```

Fig11: Loss values

7. CONCLUSION

Potholes detection is unique when compared to other object detections such as person, car, airplane and so on. Unlike other objects, potholes don't have a fixed shape. This makes it challenging for detection. In this paper we have proposed a android application to detect potholes in real-time from theimages captured by the end user. Also, in this paper we have compared to Deep Learning based SSD model for Object detection and used SSD-MobileNet Model owning to its advantages for android deployment. Both the models were trained using TensorFlow Object Detection API and MobileNet Model was integrated with our Android application using TensorFlow Lite. In the android application user details along with its location was saved into the database once the user captures the image and submits the complaint form.

8. FUTURE WORK

Our system successfully detects pothole in images but there are some works yet to be done. Those are our future works regarding pothole detection. We'll try another good CNN architectures like the latest versions of inception (inception v3 and inception ResNet) for improving the accuracy. Additionally, a GPS system has to be created for updating the location of potholes on the map for all users. In near future, we'll create a GPS enabled system which we will update the location of the pothole to the map when a user finds it on the road so that other users get the location of potholes earlier without any detection.

REFERENCES

[1] https://www.ibef.org/industry/roads-india.aspx
[2] Pothole deaths in India - iPleaders
[3] https://www.kaggle.com/atulyakumar98/potholedetection-dataset
[4] MIT World Peace University, Pune, India have proposed "Deep Learning Approach to Detect Potholes in Real-Time using Smartphone", 2019 IEEE Pune Section International Conference (Pune Con)
[5] Amita Dhiman and Reinhard Klette proposed "Pothole Detection Using Computer Vision and Learning", IEEE Transactions on Intelligent Transportation Systems (Volume: 21, Issue: 8, Aug. 2020)
[6] Pereira, V., Tamura, S., Hayamizu, S., & Fukai, H. (2018, July). A Deep Learning-Based Approach for Road Pothole Detection in Timor Leste. In 2018 IEEE International Conference on Service Operations and Logistics, and Informatics (SOLI) (pp. 279-284). IEEE.
[7] Ping Ping, Xiaohui Yang and Zeyu Gao proposed "A Deep Learning Approach for Street Pothole Detection", 2020 IEEE Sixth International Conference on Big Data Computing Service and Applications (BigDataService)
[8] Artis Mednis, Girts strazdins, reinholds zviedris, georgijs kanonirs, leo selavo Digital Signal processing laboratory,Institute of Electronics and computer science,"Real time pothole detection using android smartphones with accelerometers", 14 December str., Riga, LV 1006, Latvia .
[9] Lin, J., & Liu, Y. (2010, August). Potholes detection based on SVM in the pavement distress image. In 2010 Ninth International Symposium on Distributed Computing and Applications to Business, Engineering and Science (pp. 544547). IEEE.
[10] https://firebase.google.com/products/realtime-database?gclid=Cj0KCQjw4ImEBhDFARIsAGOTMj9JhFpHSAT2m9XtZfwXFBnjJ2f7OGUZmWbaHbGytd6lhYk1FDs3kwaAgg8EALw_wcB&gclsrc=aw.ds
[11] https://firebase.google.com/products/storage?gclid=Cj0KCQjw4ImEBhDFARIsAGOTMjYSmjnZBHYAgXmwXgyp1iIo8y2NW6nVQ_og8RUdNJmA8gcIbbpZgaAt12EALw_wcB&gclsrc=aw.ds