Creating a Semantic Segmentation Model for Pothole Detection and Optimizing it for Low-Memory Devices

Anshuman Sharma, Hibah Ihsan Muhammad, Nikhil Henry, Subham Jalan

December 15, 2023

Abstract

Contents

List of Figures

1 Introduction

In the contemporary era of self-driving vehicles and intelligent infrastructure, the accurate identification of road imperfections, such as potholes, holds paramount importance for ensuring secure and seamless transportation in India. Potholes, a ubiquitous menace on Indian roads, demand swift and accurate detection to bolster road safety and upkeep. The latest report from the transport research wing of the Ministry of Road Transport and Highways paints a concerning picture, indicating a notable rise in pothole-related accidents, deaths, and injuries. According to the report, pothole-induced accidents surged to 3,625, compared to 3,564 in 2020 (Ministry of Road Transport and Highways, 2021). Similarly, the number of associated deaths and injuries reached 1,481 and 3,103, respectively, marking an increase from 1,471 deaths and 3,064 injuries in the previous year. Shockingly, potholes accounted for nearly 0.8 percent of all road accidents, 1.4 percent of road accident-related deaths, and 0.6 percent of injuries in Million Plus Cities during 2021, underscoring the urgency of effective solutions. Our project is dedicated to employing semantic segmentation techniques tailored to precisely pinpoint and categorize potholes (Pereira et al., 2019). Moreover, we address the challenges posed by low-memory devices, commonly found in self-driving cars on Indian roads. This can also lead to creation of smaller safety-enabled systems.

1.1 Problem Statement

1. How to optimize edge detection and better classification of potholes, cracks and crevices on unpaved roads and paved roads using semantic segmentation?
2. How to optimize the model for devices of scale using various techniques such as pruning and quantization?

1.2 Objectives

1. Literature review of available convolution algorithms to process and classify road cracks and potholes.
2. Understand the application and format of image masks and determine inputs and outputs of the proposed neural network model.
3. Identify and split the image dataset into training and testing data sets.
4. Develop a model using research papers and evaluate its accuracy.
5. Learn the different methods required for optimizing the computational cost of the model like quantization and pruning.
6. Fine tune the model for better accuracy while also quantizing the model to have optimal performance on low-powered devices such as a Raspberry Pi 3 B+ for real-time detection.

Our primary objective is to synergize advanced semantic segmentation methods with optimization strategies like quantization and pruning (Cong & Xiao, 2014). This integration ensures the model operates efficiently and in real-time, even on devices with limited resources, thereby contributing significantly to the enhancement of road safety in the Indian context (Kniazieva, 2022).

2 Acknowledgements

We would like to extend our heartfelt gratitude to Dr. Nitin Upadhyaya, whose guidance and support throughout this course project have been invaluable. Your passion for the subject has truly inspired us to undertake the project.

We are also immensely thankful to the teaching assistants, Ankita Kumari, Sakshi Jaiswal, and Viraj Daniel D'souza for their tireless assistance, insightful feedback, and patience in addressing our queries. Their commitment to aiding our understanding and fostering an engaging learning environment has been fundamental to the success of this project.

Furthermore, we express our deepest appreciation to Dr. Debanga Raj Neog for their invaluable contributions and mentorship beyond the confines of this course. Their expert advice and willingness to share their knowledge have significantly enriched our project, providing a broader perspective and deeper insight into the subject matter.

Thank you all for your unwavering support and guidance.

Sincerely,

Anshuman Sharma, Hibah Ihsan Muhammad, Nikhil Henry, Subham Jalan

3 Declaration

We, Anshuman Sharma, Hibah Ihsan Muhammad, Nikhil Henry and Subham Jalan, certify that this project is our own work, based on our personal study and/or research and that we have acknowledged all material and sources used in its preparation, whether they be books, articles, reports, lecture notes, and any other kind of document, electronic or personal communication. We also certify that this project has not previously been submitted for assessment in any academic capacity, and that we have not copied in part or whole or otherwise plagiarised the work of other persons. We confirm that we have identified and declared all possible conflicts that we may have.

4 Literature Review

4.1 Artificial Neural Networks

Artificial Neural Networks can be used in a variety of applications ranging from modelling, classification and pattern recognition and aim to mimic the learning pattern present within the biological central systems in brains (Basheer & Hajmeer, 2000). Back propagation is common machine learning technique for supervised multi-layer feed forward neural networks which commonly consist of an input layer, an output layer, and one or several hidden layers (Zhu, 2018).

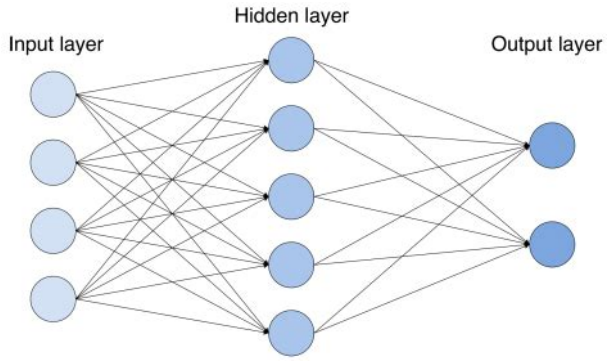
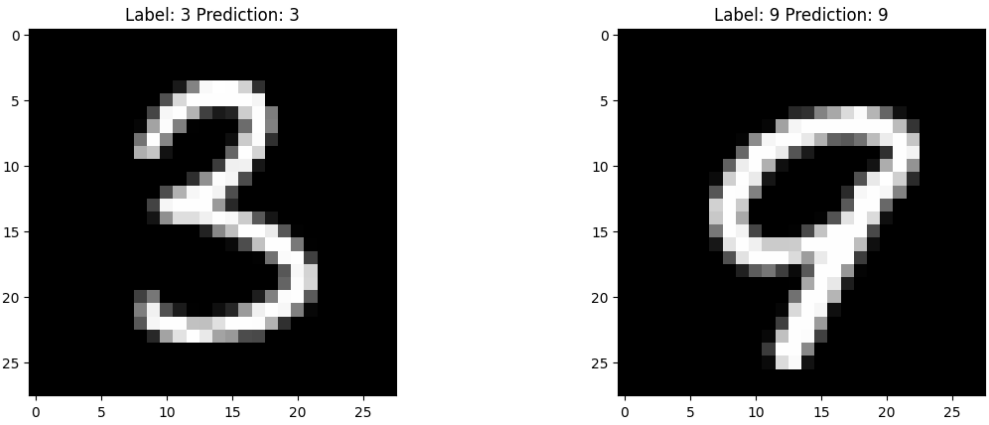


Figure 1: Back-propagation neural network structure with three layers

Figure 1 showcases the basic model of a three layer neural network. Each layer is fully connected with adjacent layer but there are not connections within a layer. Each node represents a computational unit called a perceptron which takes in inputs and applies an activation function to provide outputs. A basic perceptron consists of several elements: input, weights, activation function, threshold, and output. Weights are multiplied with inputs and then added in summing function and the sum is processed in activation function and finally, the model gives an output (Zhu, 2018).

To apply and explore the principles of a supervised back propagation neural network, the team implemented a basic 1 hidden layer classification model to classify handwritten digits ranging from 0 to 1 using the MNIST database (Modified National Institute of Standards and Technology database) by LeCun et al., 1998 using Python and numpy. The results of them same are included in Figure 2.

As introduced by LeCun et al., 2010 Convolution Neural Networks are a special kind of multi-layer neural networks, which are designed to recognise visual patterns directly from pixel images with minimal pre-processing and they can recognize patterns with extreme variability such as handwritten characters and even digits.



(a) Raw image. (b) Pothole mask applied with segmentation.

Figure 2: Output of Digit Classification Neural Net.

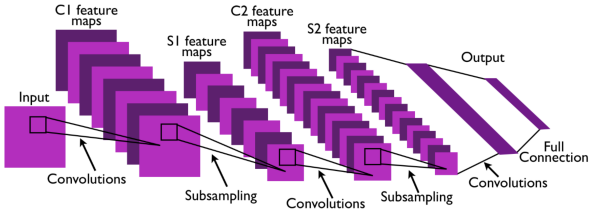


Figure 3: Overview of Convolution Neural Network.

In this project we we try to apply this principle of a convolution neural network using Keras by Chollet et al., 2015 as the underlying machine learning framework for us to build upon.

4.2 Convolutional Neural Network

Convolution is a mathematical operation that allows the merging of two sets of information. In the case of CNN, convolution is applied to the input data to filter the information and produce a feature map. Convolutional Neural Network consists of multiple layers like the input layer, convolutional layer, pooling layer, and fully connected layers. Following are descriptions of the layers that we use in our model.

- **Convolutional Layer**: This is the layer that is used to extract features from the input dataset. It applies a set of learnable filters known as kernels to the input images. The filters/kernels are smaller matrices, usually 2×2, 3×3, or 5×5 shape. It slides over the input image data and computes the dot product between kernel weight and the corresponding input image patch. The output of this layer is referred to as feature maps.

- **Max Pooling Layer**: Max pooling is a downsampling technique commonly used in CNNs to reduce the spatial dimensions of an input volume. It involves sliding a window (often called a filter or kernel) across the input data, similar to the convolution step, but instead of performing a matrix multiplication, max pooling takes the maximum value within the window. This maximum value becomes a single pixel in the new, pooled output. The window is then slid across the input data by a stride of a certain number of pixels, and the process is repeated until the entire input image has been processed.

- **Up Sampling Layer**: Up sampling is a technique that is used to increase the spatial dimension of the image. In our model, we use the Nearest Neighbour Interpolation algorithm. Every pixel in the input image is expanded to a square grid with the same values and are concatenated to create an output image.

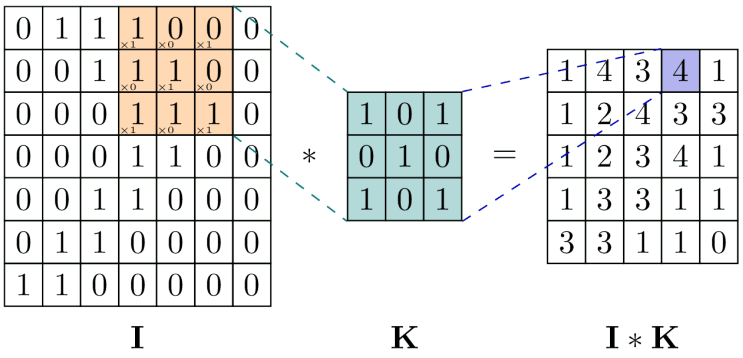


Figure 4: Convolution Layer

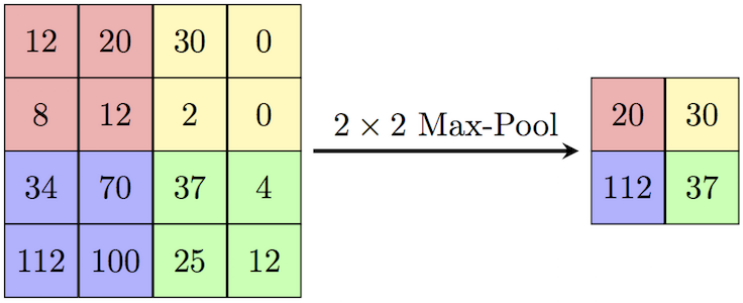


Figure 5: Max Pooling Layer

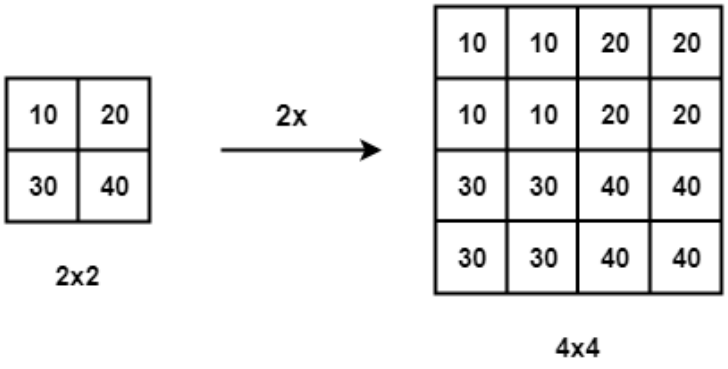


Figure 6: Nearest Neighbour Interpolation

Typically, the size of the pooling window is 2x2, and the stride with which the window is moved is also 2 pixels. This setup reduces the size of the input by half, both in height and width, effectively reducing the total number of pixels by 75%.

A Fully Convolutional Network (FCN) is a special case of a CNN that only has convolutional layers. It is also widely used for semantic segmentation. As compared to the U-Net architecture proposed below, an FCN has a few disadvantages discussed in 5.2.

4.3 Image Segmentation

Image segmentation divides images into regions with different features and extracts the regions of interest (ROIs). These regions, according to human visual perception, are meaningful and non-overlapping. There are two difficulties in image segmentation: how to define "meaningful regions", as the uncertainty of visual perception and the diversity of human comprehension lead to a lack of a clear definition of the objects, it makes image segmentation an ill-posed problem; and how to effectively represent the objects in an image. Digital images are made up of pixels that can be grouped together to make up larger sets based on their color, texture, and other information. These are referred to as "pixel sets" or "superpixels". These low-level features reflect the local attributes of the image, but it is difficult to obtain global information (e.g., shape and position) through these local attributes. The classic segmentation methods mainly focus on highlighting and obtaining the information contained in a single image, which often requires professional knowledge and human intervention. However, it is difficult to obtain high-level semantic information from images. In particular, semantic segmentation is still full of challenges due to limited or sparse annotations, class imbalance, overfitting, long training time, and gradient vanishing.

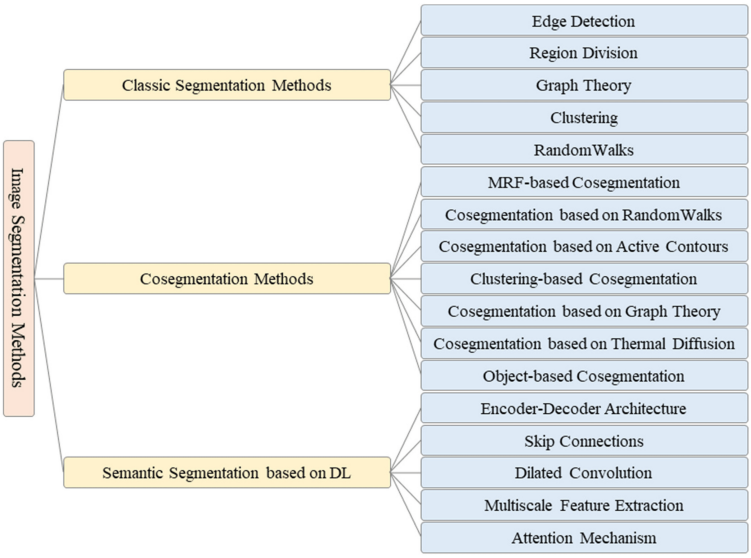


Figure 7: Categories of image segmentation methods.

4.4 Semantic Segmentation Based on Deep Learning

With the continuous development of image acquisition equipment, there has been a great increase in the complexity of image details and the difference in objects (e.g., scale, posture). Low-level features (e.g., color, brightness, and texture) are difficult to obtain good segmentation results from, and feature extraction methods based on manual or heuristic rules cannot meet the complex needs of current image segmentation, that puts forward the higher generalization ability of image segmentation models In 2015, Long et al. proposed fully convolutional networks (FCNs) with convolution instead of full connection, that made it possible to input any image size, and the FCN architecture is shown in Figure 7. FCNs prove that neural networks can perform end-to-end semantic segmentation training, laying a foundation for deep neural networks in semantic segmentation.

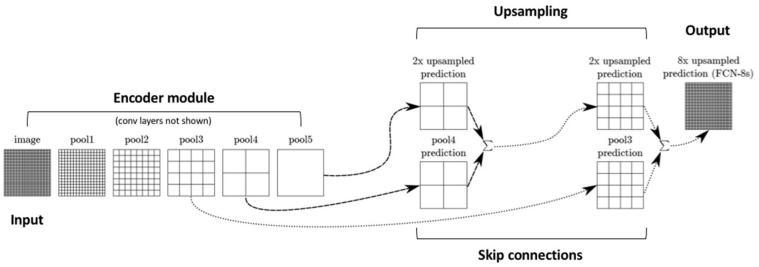


Figure 8: Overview of Fully Convolution Neural Network.

5 Methodology

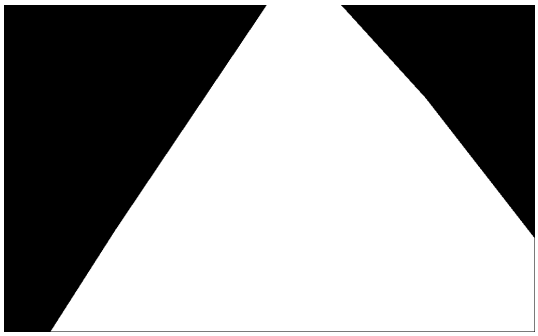
5.1 Dataset

We have used the data of paved roads in Brazil provided by Mendeley Data (Passos, 2020). The dataset contains raw images of roads, as well as their annotated masks for potholes, lanes, and cracks. However, we have only used the pothole masks in the training of our model. The database was developed using images provided by DNIT (National Department of Transport Infrastructure). It has 2235 images from highways in the states of Espirito Santo, Rio Grande do Sul and the Federal District, caught between 2014 and 2017.

(a) Raw image.

(b) Pothole mask applied with segmentation.

Figure 9: Training data set for Segmentation.



(a) Lane mask applied with segmentation.

(b) Crack mask applied with segmentation.

Figure 10: Additional segmentation masks

5.2 U-Net Architecture

We have used a U-Net neural network architecture (Ronneberger et al., 2015) for semantic segmentation of pothole images. Its layers are similar to those of an FCN (Fully Convolutional Network). However,

there are two areas where this architecture distinguishes itself.

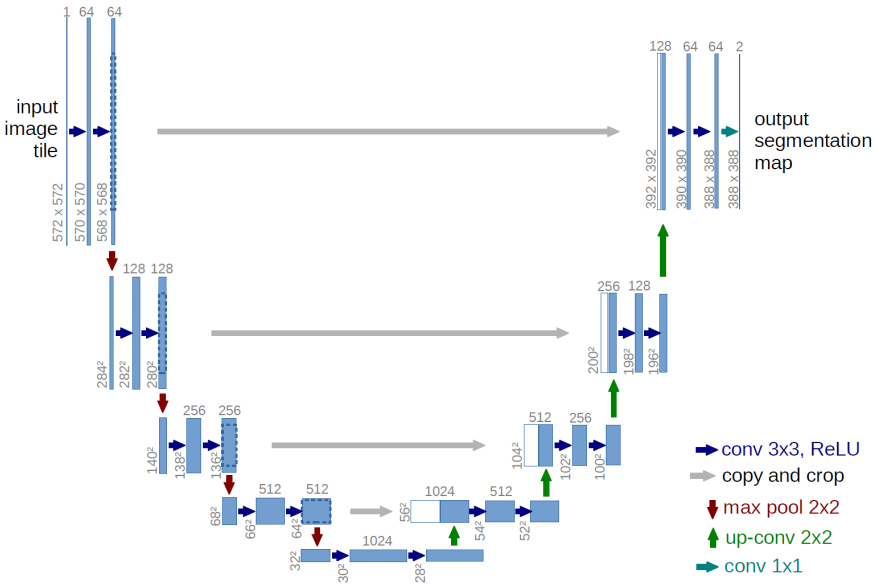


Figure 11: U-Net Architecture

First is the symmetry of the network in the encoder and decoder regions. The encoder region down samples the image using Max Pooling layers. This decreases the spatial dimension of the image and is compensated by the increase in the number of feature channels. The bridge consists of the layers connecting the encoder and the decoder region, and are responsible for classifying pixels using the given mask. The decoder up- samples the image, using Nearest Neighbour Interpolation. The second differentiating factor is the existence of skip connections. These connections allow the later layers (of the decoder) to have access to the high resolution spatial features from the earlier layers. This allows for a more accurate up-sampling of the image with the retention of localization accuracy. Otherwise, the decoder would not have been able to classify all the pixels of the detected object in the output image.

1. **Precise Object Localization**: U-Net incorporates skip connections that allow the network to merge low-level and high-level features, aiding in precise object localization by preserving fine details and context information which FCNs might lose during downsampling and upsampling.

2. **Enhanced Boundary Detail**: The architecture of U-Net, with skip connections and its 'U' shape, facilitates better preservation of object boundaries, enabling sharper and more accurate delineation of object edges compared to FCNs, which might produce blurrier boundaries.

3. **Effective Feature Reuse**: U-Net's design enables efficient reuse of features through skip connections, reducing the risk of information loss and optimizing the network's ability to learn and use relevant information from different scales, resulting in improved segmentation performance compared to FCNs.

We use the ReLU activation function in the hidden layers, which is defined as follows.

$$f(x_i) = max(0, x_i)$$

An activation function assists in adding non-linearity to the neuron of a network. ReLU is desirable in place of other functions (like Sigmoid) to solve the problem of vanishing gradient. Vanishing gradient refers to a phenomenon where the gradients of the activation functions approach zero as we back propagate through the network for their computation.

The architecture primarily uses the Cross-Entropy loss function, which is defined as below.

$$L = \frac{-1}{|\Omega|} \sum_{x_i \in \Omega} \log p_{y_i}(x) \tag{1}$$

where:

- Ω: set of all pixels
- y_i ground truth of x_i
- $\log(p_{y_i}(x))$: prediction that x_i is of class y_i

We have trained the model with a variety of different loss functions and have compared their results.

5.3 Model Development

The implementation of the final model took place in three distinct phases:

1. **Data Pre-processing**: Before initiating model training, the acquired dataset underwent a pre-processing phase. The dataset exclusively comprised a singular class of potholes, encompassing allied cracks and crevices, alongside their corresponding masks. Each image within the dataset may or may not contain a pothole. The associated masks were binary images where pixels unrelated to potholes were deactivated or represented by black pixels. Potholes, the targeted class for identification, were activated and represented by white pixels. To streamline image dimensions and reduce computational resources during evaluation, images were converted to grayscale and subsequently downscaled to 256x256 pixels. After conducting experiments within our scope, we concluded that utilizing an image size of 256x256 yielded superior results compared to an image size of 128x128.

2. **Model Definition**: The approach employed for model development involved leveraging transfer learning to ascertain its efficacy. Transfer learning, a machine learning technique, utilizes knowledge gained from one task to enhance performance in a related task. In our specific scenario, we utilized the 'resnet34' pre-trained model, extensively trained on the Imagenet2012 dataset, comprising over 100,000 images across 200+ classes. The transfer learning method expedites convergence and offers accuracy conducive to evaluating alignment with our defined objectives. By initializing the model with 'imagenet' encoder weights, we secured the most accurate parameters necessary for image classification. Subsequently, our model adjusted these parameters to accommodate binary semantic segmentation requirements. To achieve this, we implemented the U-net architecture as objects using the 'segmentation models' Python library, equipped with neural networks tailored for image segmentation via TensorFlow and Keras. For parameter optimization, we selected Adam's optimizer and employed the binary cross-entropy loss function to guide the optimization process.

3. **Model Fitting**: The model underwent training for 100 epochs in batch sizes of 50 during this phase.

5.4 Loss Functions

Within machine learning, a loss function serves as a mathematical model evaluating the disparity between the current output of a model and the expected output. It functions as a metric measuring model performance, providing crucial feedback to guide the optimization process, thereby assessing the model's alignment with the provided dataset and predefined objectives. The following loss functions were employed in our project to enhance the efficiency of delineating pothole boundaries:

- **Binary Cross Entropy** is defined as a measure of the difference between two probability distributions for a given random variable or set of events. It is widely used for classification objective, and as segmentation is pixel level classification it works well (Jadon, 2020). Binary Cross-Entropy is defined as

$$L_{\text{BCE}}(y, \hat{y}) = -[y \log(\hat{y}) + (1 - y) \log(1 - \hat{y})] \tag{2}$$

where:

- $\hat{y}$: is the predicted value by the model
- y is the expected output value from the training dataset x

- **Focal Loss**: A variant of binary cross entropy, focal loss assigns weights to different examples, accentuating learning from challenging instances while mitigating the influence of easier examples. It notably addresses scenarios characterized by significant class imbalances, such as our case, where the background (comprising elements other than potholes) dominates as the majority class, while the foreground (encompassing potholes and related features) constitutes a minority class. It Loss proposes to down-weight easy examples and focus training on hard negatives using a modulating factor, as shown below from Jadon, 2020.

$$FL(p_t) = -\alpha_t(1-p_t)^{\gamma}\log(p_t) \tag{3}$$

where:

- p_t: is the probability that a pixel belongs to that said class.
- α_t and γ are hyper parameters that need to be tuned while training the model.

- **Dice Loss**: This loss function quantifies the discrepancy in overlap between predicted and targeted segmentation masks. It offers a smooth and differentiable measure of segmentation accuracy, particularly beneficial when the objective revolves around capturing intricate details within segmentation masks. The Dice coefficient is widely used metric in computer vision community to calculate the similarity between two images. Later in 2016, it was also adapted as a loss function (Sudre et al., 2017).

$$DL(y,\hat{p}) = 1 - \frac{2y\hat{p}+1}{y+\hat{p}+1} \tag{4}$$

where:

- $\hat{p}$: is the predicted probability that a pixel belongs to that said class.
- y the correct class of a pixel from the training dataset.

Implementation of these diverse loss functions has been derived from the Keras library, serving the specific objectives of this project. The results stemming from the utilization of these loss functions are discussed upon in the results section of this report. Our metric for evaluating the model revolves around its accuracy achieved through the utilization of these aforementioned loss functions.

5.5 Quantization

Quantization represents an optimization technique employed to decrease the precision of the parameters within a machine learning model. Its primary function involves converting model weights from high-precision floating-point values to low-precision integers, typically 16 or 8 bits. This reduction in weight parameter precision serves to minimize storage requirements and enhance computational efficiency in processing mathematical operations, especially in comparison to high-precision float values. Additionally, it proves advantageous for optimizing devices at the edge scale lacking high-speed processors, notably embedded systems.

There exist two principal methods for quantizing a model:

1. **Quantization Aware Training**: This method involves computing the range for model weights and activations during the training phase, integrating the quantization process into the training process itself. By doing so, the model can retain its accuracy while benefiting from the advantages offered by quantization.

2. **Post Training Quantization**: Post Training Quantization converts the weights and activations of a pre-trained network from float 32 representation to 8-bit integers. This conversion notably reduces memory requirements and computational resources with minimal degradation in model accuracy.

In our project, 'Post Training Quantization' using Dynamic Range Quantization is employed to downscale the model. Leveraging TensorFlow libraries, this method scales down the model size by approximately fourfold. Essentially, it modifies the range of model parameters to possess 8-bit integer precision, facilitating testing of the model's functionality on edge scale devices, aligning with our project's objectives.

The specific implications and outcomes of this segment of our project are yet to be thoroughly explored. However, upon recognizing the efficacy of these methods, we aim to incorporate them to broaden the project's scope.

Initially, our model boasted over 2,440,000 trainable parameters, resulting in a model size of approximately 93.29 MB. Implementing Dynamic Range Quantization notably reduced the model size by around 3.88 times, resulting in a quantized model size of approximately 24 MB. As for the accuracy of the model concerning the provided dataset and its performance with different loss functions, this aspect remains unexplored territory. Nevertheless, we remain optimistic about achieving a similar level of reliability as in the original model through further exploration and experimentation.

6 Results

We consolidated our research findings into both GitHub repository and Google Collab Notebooks.

We found using TensorFlow through Keras to provide us with the most consistent output and results over multiple training epochs.

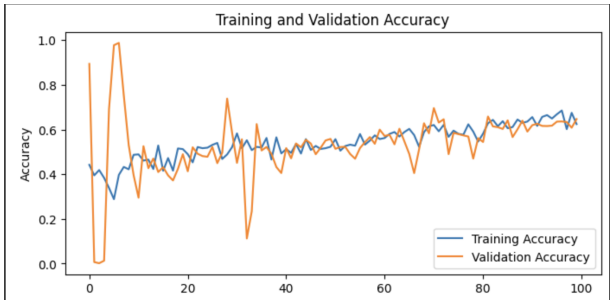


Figure 12: Plot of Accuracy v/s Epochs

From the above plot in Figure 12, we can see that our model reached an accuracy of 0.7 after training for 100 epochs. The rapid peaks present within the plot are indicative of numerous false positives present in the dataset. The models struggles to converge at these points yet as the number of completed training epochs increased the model was slowly able to dampen the influence and hence was able to correctly infer the not only the location but also the whether or not a pothole itself was present within the dataset.

Now comparing the output of the model with reference to the input image.

(a) Raw image.

(b) Pothole mask applied with segmentation.

Figure 13: Output of Model Post Training

References

Basheer, I. A., & Hajmeer, M. (2000). Artificial neural networks: Fundamentals, computing, design, and application. *Journal of microbiological methods*, *43*(1), 3–31.

Chollet, F., et al. (2015). Keras.

Cong, J., & Xiao, B. (2014). Minimizing Computation in Convolutional Neural Networks. In S. Wermter, C. Weber, W. Duch, T. Honkela, P. Koprinkova-Hristova, S. Magg, G. Palm, & A. E. P. Villa (Eds.), *Artificial Neural Networks and Machine Learning – ICANN 2014* (pp. 281–290). Springer International Publishing. https://doi.org/10.1007/978-3-319-11179-7_36

Jadon, S. (2020). A survey of loss functions for semantic segmentation. *2020 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)*. https://doi.org/10.1109/cibcb48159.2020.9277638

Kniazieva, Y. (2022, September). Semantic Segmentation. Retrieved October 30, 2023, from https://labelyourdata.com/articles/semantic-segmentation

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2324.

LeCun, Y., Kavukcuoglu, K., & Farabet, C. (2010). Convolutional networks and applications in vision. *Proceedings of 2010 IEEE international symposium on circuits and systems*, 253–256.

Ministry of Road Transport and Highways. (2021, February). Road Accidents in India - 2021 — Ministry of Road Transport & Highways, Government of India. Retrieved October 30, 2023, from https://morth.gov.in/road-accidents-india-2021

Passos, B. T. (2020). Cracks and potholes in road images.

Pereira, V., Tamura, S., Hayamizu, S., & Fukai, H. (2019). Semantic Segmentation of Paved Road and Pothole Image Using U-Net Architecture. *2019 International Conference of Advanced Informatics: Concepts, Theory and Applications (ICAICTA)*, 1–4. https://doi.org/10.1109/ICAICTA.2019.8904105

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. *CoRR*, *abs/1505.04597*. http://arxiv.org/abs/1505.04597

Sudre, C. H., Li, W., Vercauteren, T., Ourselin, S., & Jorge Cardoso, M. (2017). Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations. In *Lecture notes in computer science* (pp. 240–248). Springer International Publishing. https://doi.org/10.1007/978-3-319-67558-9_28

Zhu, W. (2018). Classification of mnist handwritten digit database using neural network. *Proceedings of the research school of computer science. Australian National University, Acton, ACT*, *2601*.

7 Appendix

7.1 U-Net Implementation in PyTorch

Here is the link to our GitHub repository: Pothole Segmentation
This model requries weight training and does not utilize transfer learning. On two Nvidia A4000 GPUs, the model takes approximately 17 hours to train. The model did not yield good results and the high training time did not give us enough opportunity to figure out its failure points, use different loss functions and improve the accuracy of the model. We used Cross Entropy in the initial training of this model.

7.2 Colab Notebook

Here is the link to our Colab Notebook: Colab Notebook
This notebook contains the code that was used for predicting the masks that have been mentioned in the results section. We are using pre-trained weights of the resnet34 backbone in TensorFlow Keras. The code can be modified to train the model with different loss functions and compare the results.