Pavement Crack Detection and Segmentation Using Deep Learning

INDIAN INSTITUTE OF TECHNOLOGY KANPUR

Session 2020-21 I

CE784A Project Presentation

Pavement Crack Detection and Segmentation Using Deep Learning

Professor: Pranamesh Chakraborty, Department of Civil Engineering

Group Brogrammers
Aastha Ramteke (180009)
Akarsh Mittal (180051)
Lakshay Middha (180377)
Naman Jain (180459)
Neha Aggarwal (180466)

Motivation

- **Cracks** are fracture lines caused due to temperature change, moisture expansion, elastic deformation, settlement, or creep.
- Due to intensive use of roads, cracks arise in the pavements, which can cause serious damage in extreme conditions.
- Road traffic is increasing day by day, resulting in increase in cost of road maintenance. So it's important to detect defaults before repair costs are too high.
- Traditional crack detection methods depend mainly on manual work and are limited by the following:
    - they are time consuming and laborious
    - they rely entirely on human experience and judgment.

Introduction

- Our paper proposes **Pavement Crack Detection and Segmentation using Deep Learning** solution of the problem.
- First, images are **classified** as cracked or uncracked using different SOTA models.
- Then, we ran a pixel level **segmentation** model to generate annotations of the cracked images.
- The method we proposed can help to monitor road distress quality and early detection of damages in the pavements.
- This idea can be extended to structural damage detection in buildings or detecting cracks in vehicle tires too.

Review of Traditional Methods

- The development of this technology can be parted into 3 stages -
  - Traditional manual detection
  - Semi-automatic detection
  - Full automation
- Machine Learning uses feature extraction and pattern recognition for crack detection. Many scholars worked on crack detection including
  - Oliveira et al. use the mean and the standard deviation for unsupervised learning to distinguish blocks with crack from blocks without crack.
  - In CrackForest, Shi et al. proposed a new descriptor based on random structured forests to characterize cracks.
  - Cha et al. use a sliding window to divide the image into blocks and CNN is used to predict whether the block contains cracks or not.

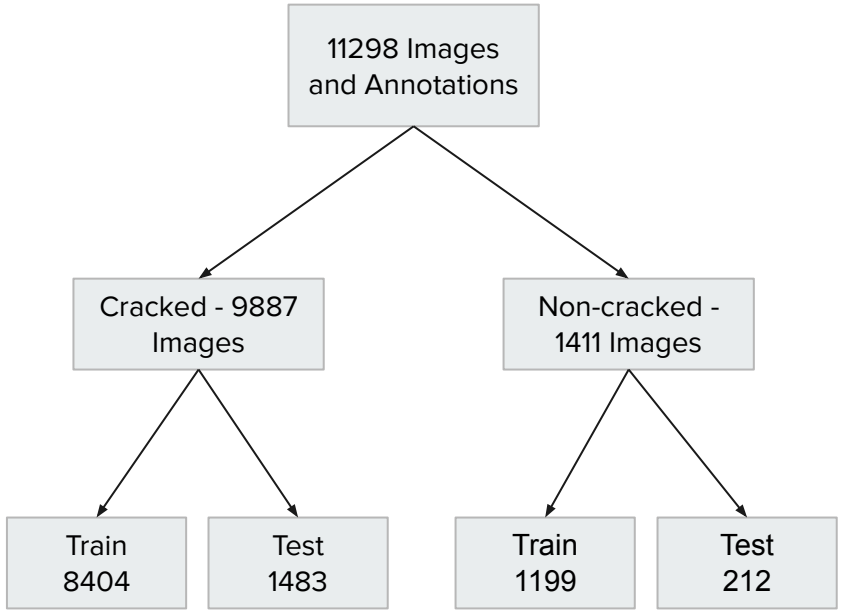
Review of New Methods

- New methods involve **segmentation** which helps us to localize the crack and obtain comprehensive crack characteristics.
- There are 3 broad levels of automatic pavement crack detection.
    - Image-Level Crack Detection
    - Block-Level Crack Detection
    - Pixel-Level or Pixel-Wise Crack Detection
- Zhang et al. (2016b) proposed a CNN which outperformed traditional machine learning techniques (i.e., SVM and Boosting method).
- Cha et al. (2017) developed a deep CNN for piecewise classification on cracks but the pixel-perfect accuracy was unattainable.
- SegNet (2017) proposed an idea of the encoder-decoder architecture.

Methodology

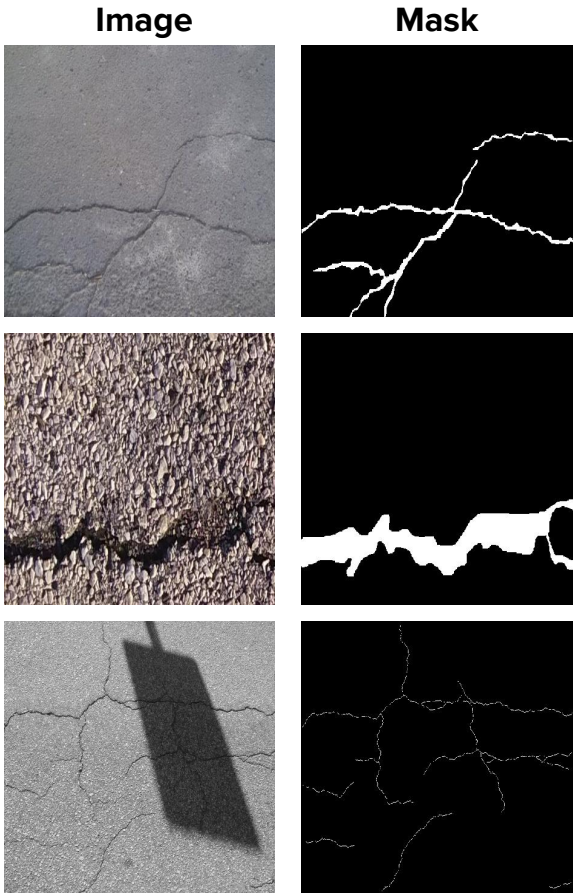
Dataset Details

Image Classification and Segmentation



Dataset Details

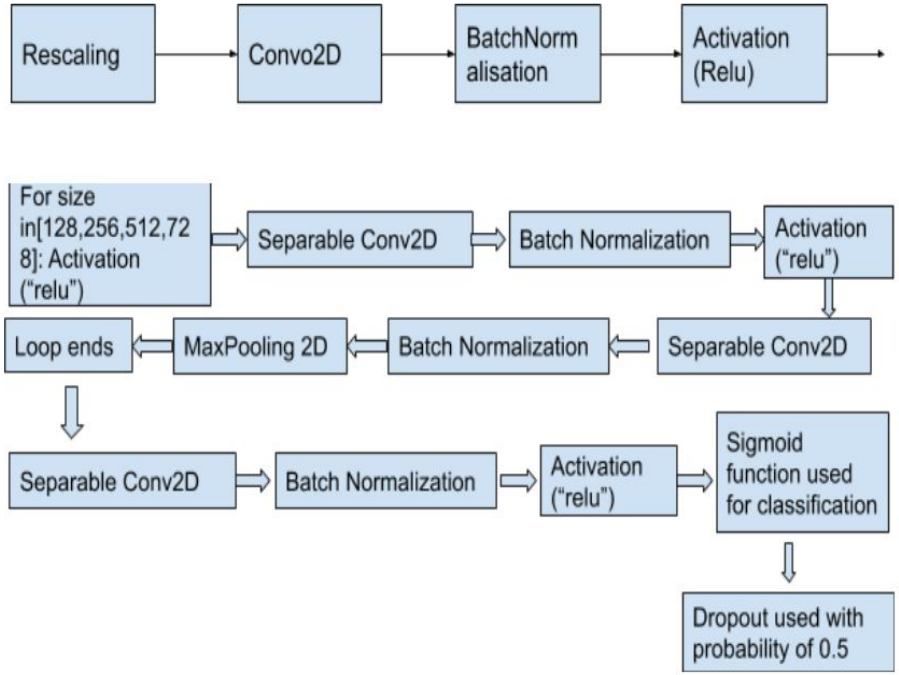
Image Classification and Segmentation



Crack Classification

Model-1 Architecture

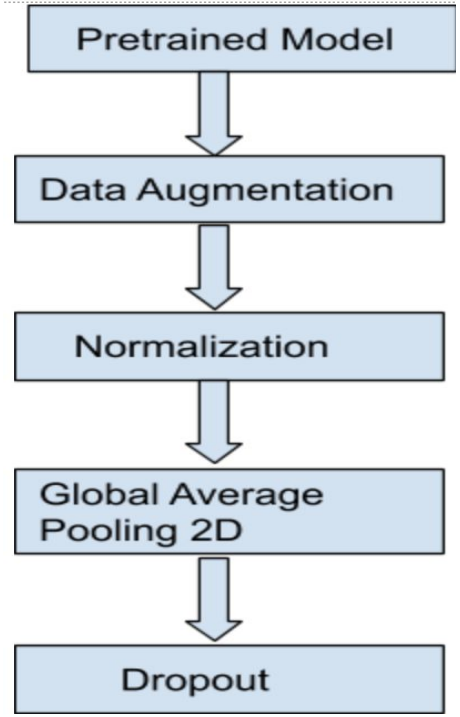
- Input images were rescaled and then fed into a series of convolutional block which consists of convolutional layers, batch normalization and then applying activation ReLU.
- Final feature map is passed through sigmoid activation function to generate probabilities and then results are obtained by using dropout layer with probability **0.5.**



Inception V3

- Images were resized and fed into the model.
- Feature maps from pre-trained **Inception V3** model were used.
- Then the maps were augmented, normalized, passed through average pooling layer and then a drop out layer to predict the final result.

```
base_model = tf.keras.applications.InceptionV3(
    include_top=False,
    weights="imagenet",
    input_shape=(299, 299, 3),
)
```

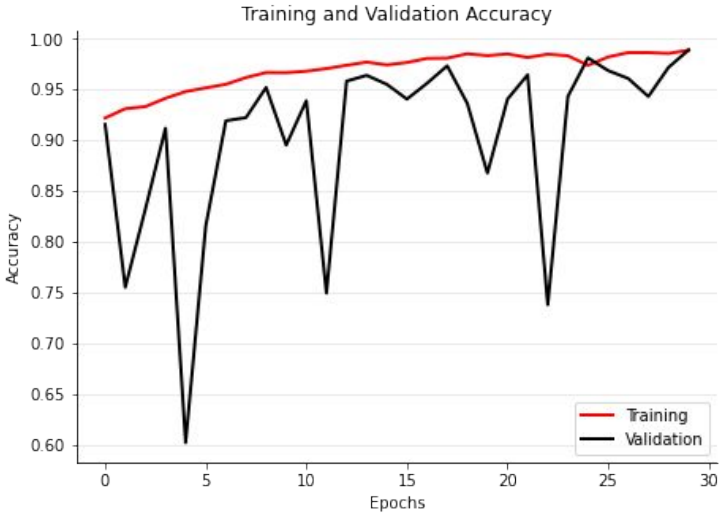


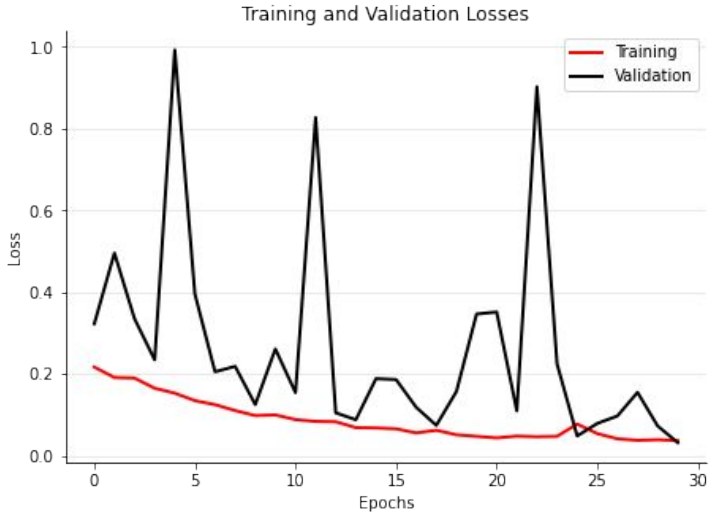
Training

- Training was done for **8404** images while testing was done for **1483** images on Nvidia K80 GPU available on Kaggle.
- Better results were obtained using the pretrained Inception V3 model and then fine tuning it in comparison with the model build from scratch (Model 1).
- Model 1 achieved an accuracy of **97.9%** after being trained for 30 epochs, and time per epoch was around 90 seconds.
- Inception V3 model achieved an accuracy of **99.9%** after being trained for 7 epochs, and time per epoch was around 60 seconds.

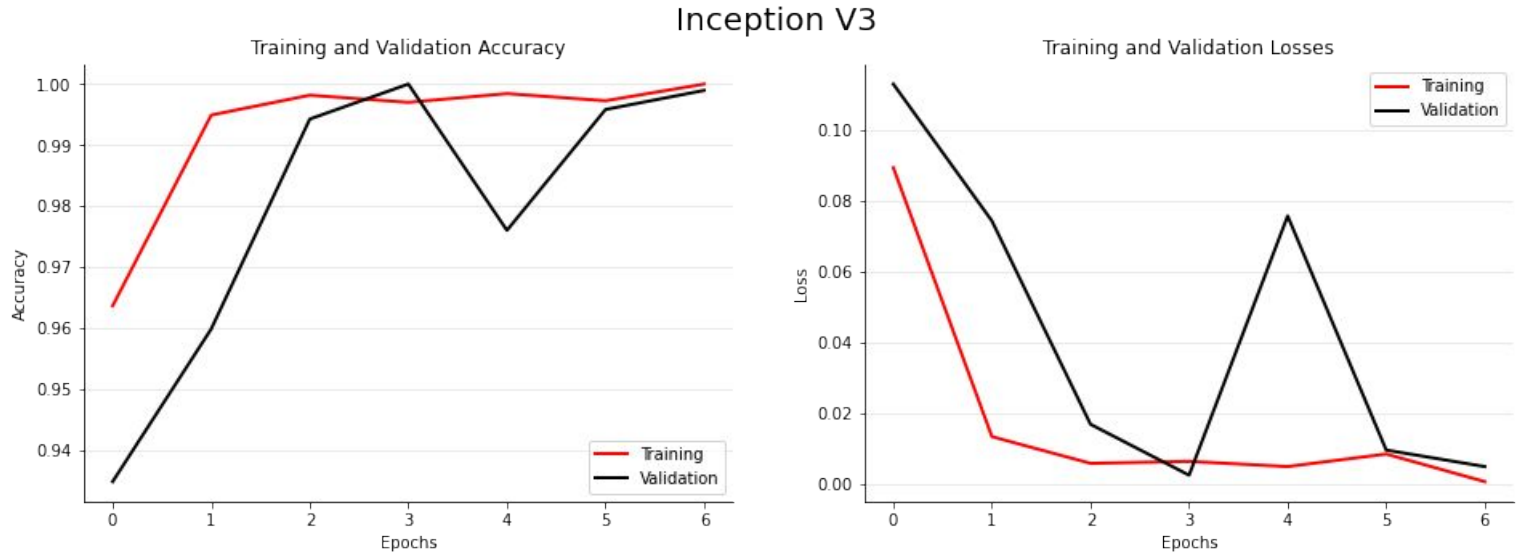
Training

Model 1





Training


Inception V3
Training and Validation Accuracy
Accuracy
Epochs
Training
Validation
Training and Validation Losses
Loss
Epochs
Training
Validation

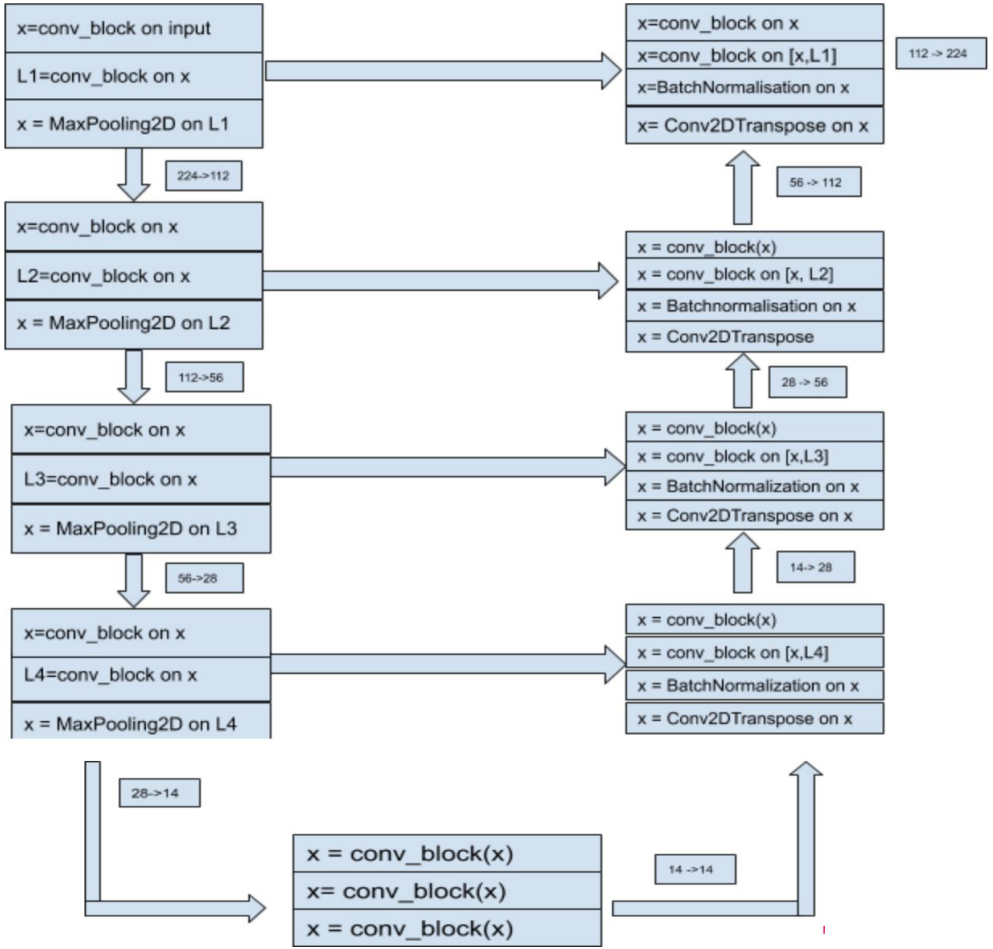
Testing and Results

Model	Cracked				Uncracked				Total				Accuracy
	TP	TN	FP	FN	TP	TN	FP	FN	TP	TN	FP	FN	
Model1	1465	0	18	0	0	196	0	16	1465	196	18	16	**0.979**
Xception	1481	0	2	0	0	204	0	8	1481	204	2	8	**0.994**
Inception V3	1481	0	2	0	0	211	0	1	1481	211	2	1	**0.998**

Crack Segmentation

U-Net Architecture

- Consists of **encoder** and **decoder** layers
- Encoding layers extract features using convolutional and pooling operations
- Decoding layers use transpose convolution along with side outputs to generate sharp features
- Final upsampled feature maps are concat -enated and a threshold is set to generate annotated image



Unet Architecture

- Left block:

```
x = conv_block(inputs, 32, (3, 3), strides=1, name='L_conv1-1')
L1 = conv_block(x, 32, (3, 3), strides=1, name='L_conv1-2')
x = MaxPooling2D((2, 2), strides=(2, 2), padding='same')(L1)
```

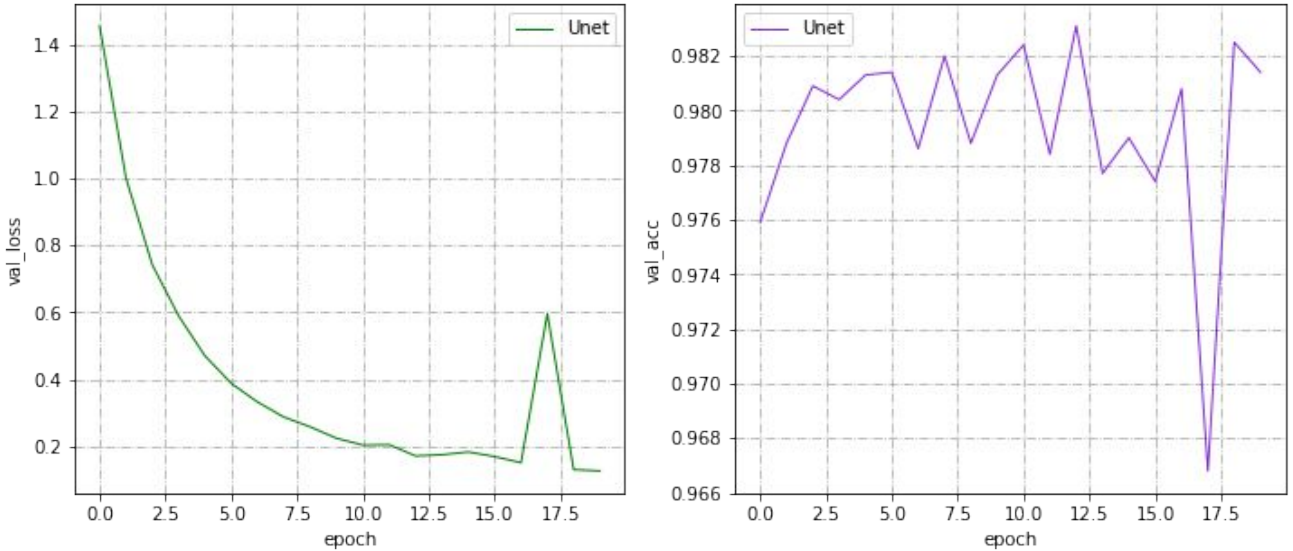
- Right block

```
x = Conv2DTranspose(32, kernel_size=2, strides=2, padding='same', name='R_conv4-1')(x)
x = BatchNormalization(axis=bn_axis, name='R_conv4-1_' + 'bn')(x)
x = conv_block(Concatenate(axis=-1)([x, L1]), 32, (3, 3), strides=1, name='R_conv4-2')
x = conv_block(x, 32, (3, 3), strides=1, name='R_conv4-3')
```

Training

- Training was performed for cracked images (**8404** images) and their annotations using Nvidia K80/T4 GPU of Google Colab.
- Total training time was greater than 2 hours for a total of **20** epochs.
- Test images (around **1400**) of the dataset and their annotations were used as validation data
- Learning rate was reduced by a factor of **5** by monitoring the validation loss.
- Model performed with an accuracy of **98.24%** on the training data and **98.14%** of accuracy on validation data.

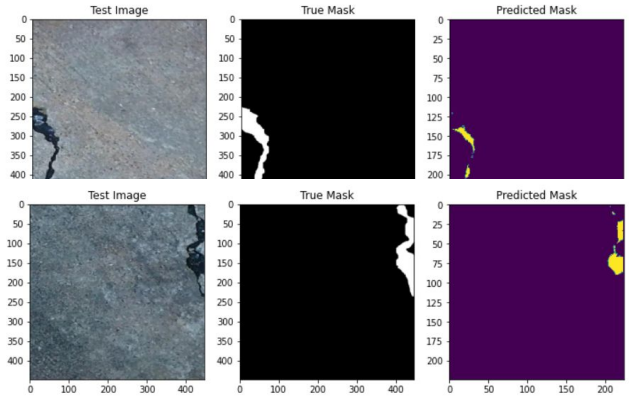
Training

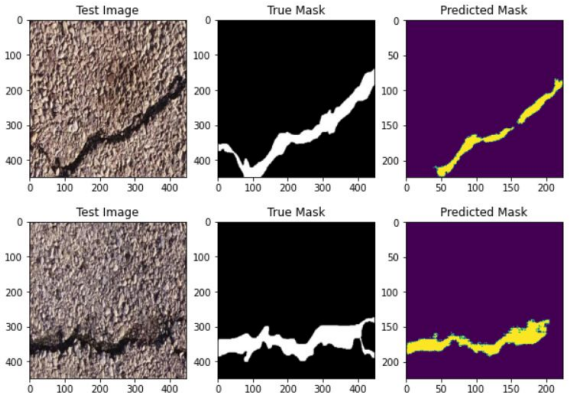

Unet
val_loss
epoch
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
Unet
val_acc
epoch
0.982
0.980
0.978
0.976
0.974
0.972
0.970
0.968
0.966

Testing and Results

- For generating segmentation masks, prediction was made on the validation dataset and it took a total time of **15 minutes.**
- The following results were obtained:

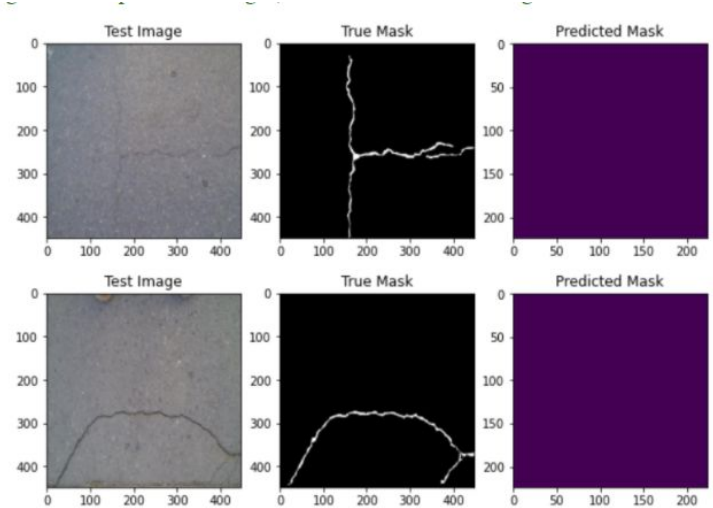
.





Testing and Results

- Initially, the predictions seemed pretty good, but after visualizing some more images, the following results were also observed:

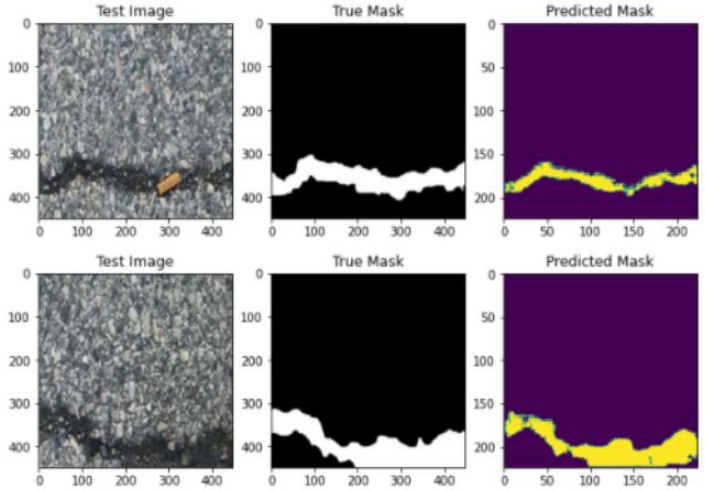


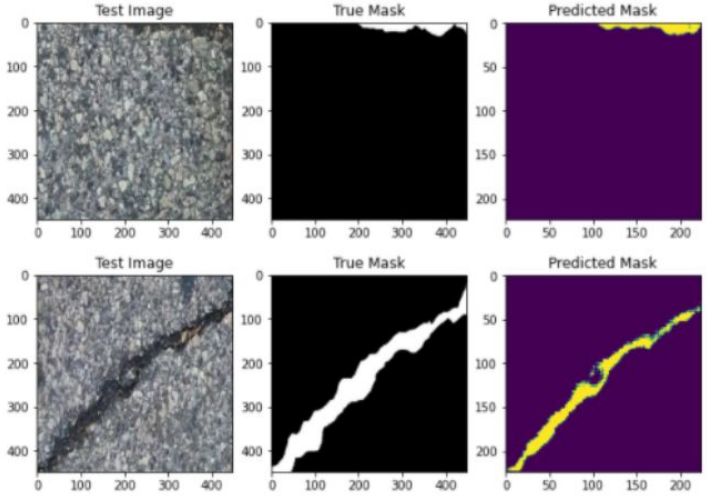
Testing and Results

- Mean Intersection-over-Union (IOU) ratio and F1 score for the validation dataset were **very low**.
- Model failed to generate annotations for different varieties of images in the dataset.
- Model performed good only for Crack500 images (IOU near **0.5** and f1_score > **0.6**) while it could not perform well for other datasets like Crack Forest, CrackTree resulting in an IOU and F1 score of **~0**.

Testing and Results

- Final prediction were made for **504** images of Crack500 dataset and the results were:





Testing and Results

- For some images IOU was greater than **0.5** and F1 score was around **0.6.**
- But for some images IOU and F1 score **were low.**
- Hence, mean IOU came out to be **0.17** while mean F1 score was **0.25.** (the results were skewed due to poor predictions on some particular images)

Limitations of the Model

- Annotations could not be generated properly for some of the images in the validation dataset.
- For some images, annotations were generated , but the IOU was only around **0.5.**
- Training was only done for **20** epochs. If trained more, might result in good results.
- Reducing image size as per input shape might also be a factor for loss of certain features of the image.

Conclusion

- Dataset of more than **11000** images with their annotations were used in this research.
- Crack Classification was achieved with an accuracy of **99.5%** using Transfer Learning technique.
- For segmentation, only cracked images were used for training and testing.
- Used Encoder-Decoder model to perform image segmentation task.
- Model did not perform as per expectations and F1 score on training data was very low.
- Final predictions were made on **504** images of crack500 dataset and the mean iou score was **0.17** while the mean f1_score was **0.25.**

Further Improvements

- Although the results were fairly good, the number of epochs could be increased to obtain better accuracy scores.
- The research can be extended to training more advanced pavement crack segmentation models like **Deep Crack**, **Crack U-Net**.
- Pre trained weights could also be used, to make the model learn better.

References

- Dataset: https://github.com/khanhha/crack_segmentation#dataset
- A. Zhang, K. C. Wang, B. Li, E. Yang, X. Dai, Y. Peng, Y. Fei, Y. Liu, J. Q. Li, and C. Chen, “Automated pixel-level pavement crack detection on 3D asphalt surfaces using a deep-learning network,” Computer-Aided Civil and Infrastructure Engineering, vol. 32, no. 10, pp. 805–819, 2017.
- Yahui Liu, JianYao, XiaohuLu, Renping Xie, LiLi : “DeepCrack: A deep hierarchical feature learning architecture for crack segmentation“
- Ju Huyan, Wei Li, Susan Tighe, Zhengchao Xu, Junzhi Zhai : ”CrackU-net: A novel deep convolutional neural network for pixelwise pavement crack detection”
- H. Oliveira and P. L. Correia, “Automatic road crack detection and characterization,” IEEE Transactions on Intelligent Transportation Systems,vol. 14, no. 1, pp. 155–168, 2013.

Thank You