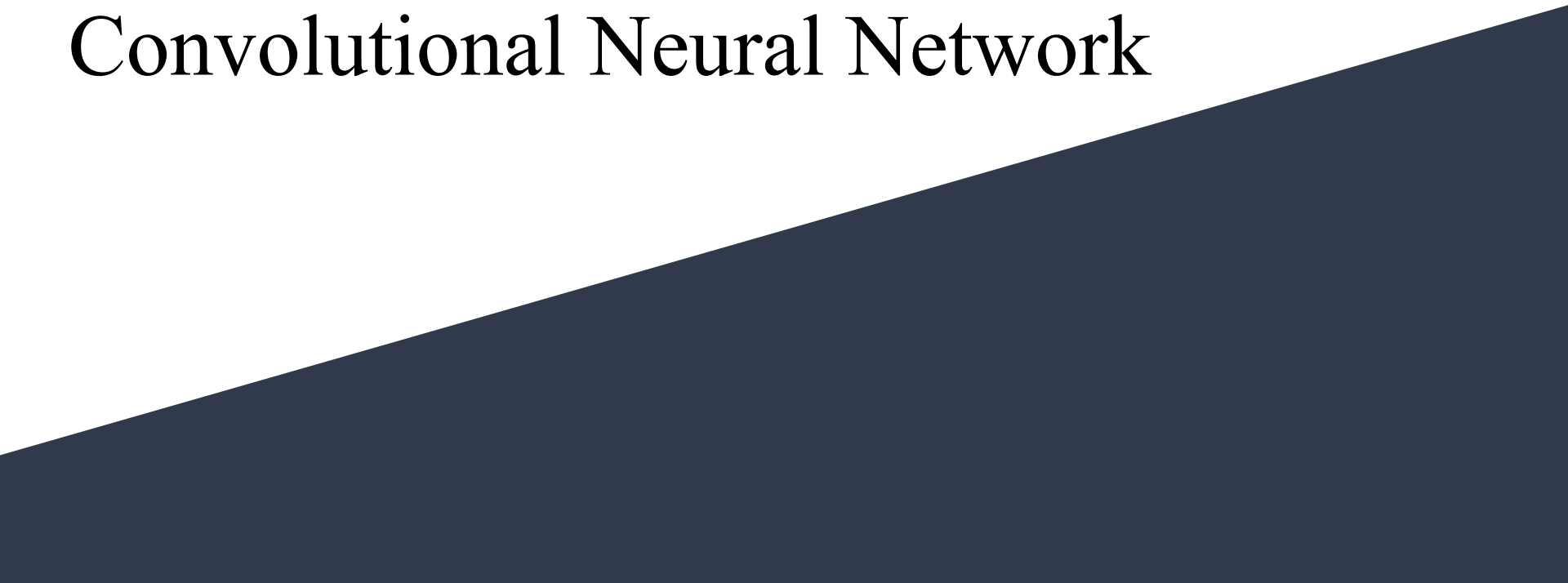


# Surface Crack Detection Using Deep Convolutional Neural Network

A dark blue diagonal gradient bar that starts from the bottom left and extends towards the top right, covering the lower half of the slide.

# Rajshahi University Of Engineering & Technology

## Department of Computer Science & Engineering

Presented By

Sadman Sakib Radh

Roll: 1503007

Dept. of CSE, RUET

Supervised By

Prof. Dr. Md. Al Mamun

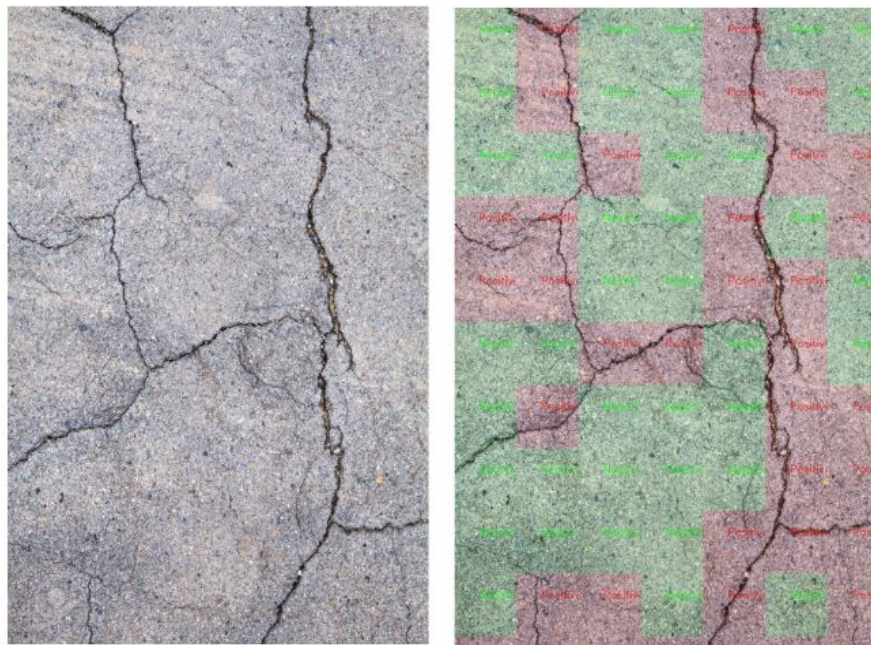
Professor

Dept. of CSE, RUET

# Outline

1. Introduction
2. Motivation
3. Literature Review
4. Dataset
5. Methodology
6. Implementation
7. Result Analysis
8. Conclusion

# Introduction



Measurement of the extent of cracks on the surface of a material (usually a solid material such as metals, plastics and ceramics).

# Motivation

1. Concrete surface cracks are major defect in civil structures
2. Detecting Crack of surfaces more accurately
3. More accuracy ensures less accident in highway
4. Inspecting, finding the cracks and determining the building health

# Literature Review

1. Autonomous concrete crack detection using deep fully convolutional neural network [2]

Applied Method - VGG16, Inception V3, ResNet50

Dataset - 40,000

Accuracy - 0.998, 0.997, 0.975

# Literature Review

## 2. Road Crack Detection Using Deep Convolutional Neural Network [3]

Applied Method - ConvNets

Dataset - 40,000

Accuracy - 0.8965

# Dataset



Fig 2: Crack Image [3]

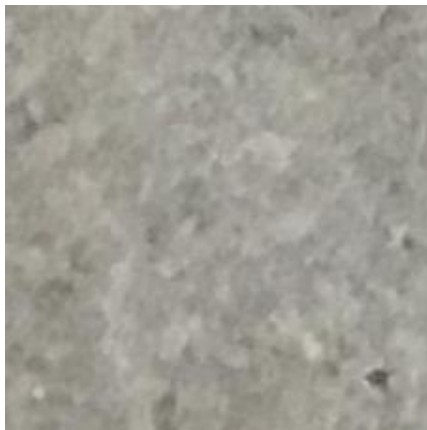


Fig 3: Non Crack Image  
[3]

- The dataset is created from 458 (4032x3024 pixel) high-resolution images using the method proposed by Zhang et al (2016) [3]
- Consists of 40,000 images of 227 x 227 in RGB
- 20,000 positive images (Crack)
- 20,000 negative images (Non Crack)



# Methodology

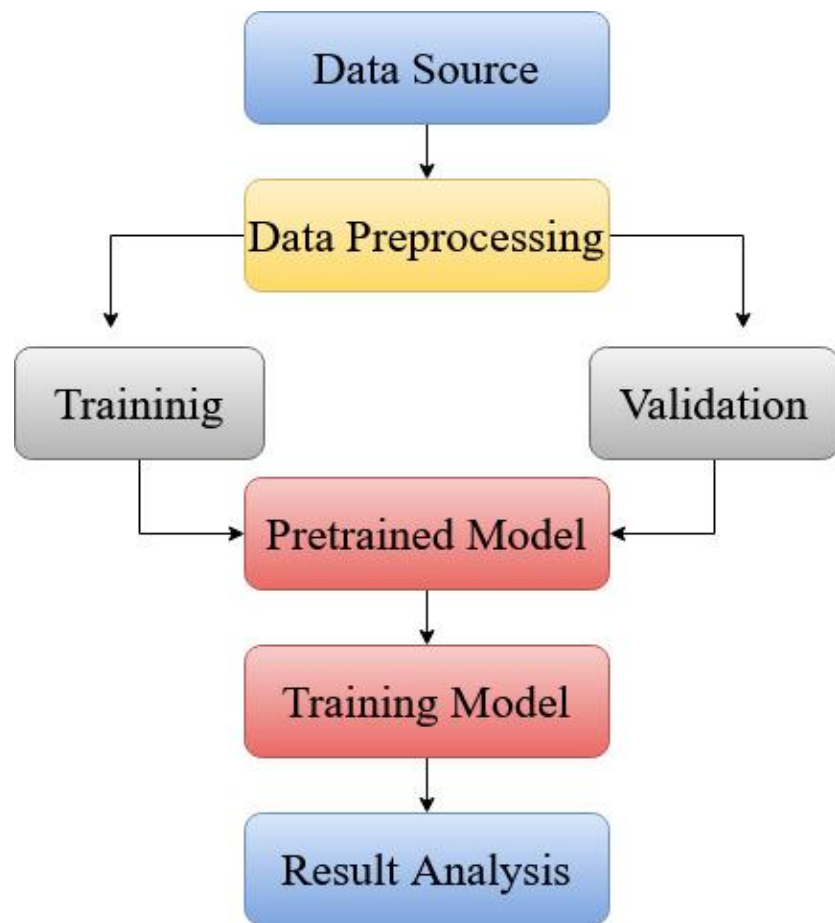


Fig 4: Workflow

# Implementation

## 1. Data Preprocessing:

- (a) Image Resize: Images are resized into 150 x 150 (RGB) to make all image size uniform for VGG16 model [4] (pretrained model)
- (b) Normalization: Images are rescaled between 0 to 1 by dividing each pixel value by 255

## 2. Data Split:

Datasets are splitted into 'Training' and 'Validation' (30 percent randomly) sets

# Implementation

## 3. Model Architecture:

VGG-16



Fig 5: VGG16 Model [4]

# Implementation

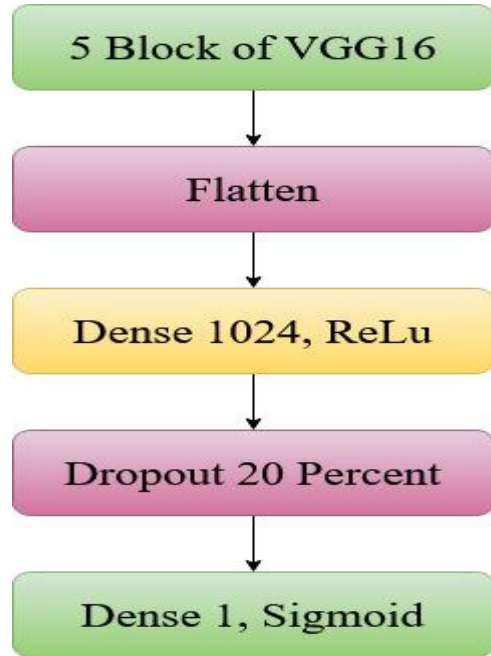


Fig 6: Model Architecture

## Parameters:

Epoch: 20

Batch Size: 64

Optimizer: Adam [5]

Loss Function: binary\_crossentropy [6]

Dropout: 20 percent

Learning rate: 0.001

# Result Analysis

```
Epoch 19/20
438/438 [=====] - 71s 162ms/step - loss: 0.0044 - accuracy: 0.9986 - f1_m: 0.9986 - precision_m: 0.9987 - recall_m: 0.9986 - val_loss: 0.0161 - val_accuracy: 0.9973 - val_f1_m: 0.9972 - val_precision_m: 0.9975 - val_recall_m: 0.9971
Epoch 20/20
438/438 [=====] - ETA: 0s - loss: 0.0017 - accuracy: 0.9994 - f1_m: 0.9994 - precision_m: 0.9994 - recall_m: 0.9993
Reached 99.9% accuracy so cancelling training!
438/438 [=====] - 70s 160ms/step - loss: 0.0017 - accuracy: 0.9994 - f1_m: 0.9994 - precision_m: 0.9994 - recall_m: 0.9993 - val_loss: 0.0161 - val_accuracy: 0.9974 - val_f1_m: 0.9974 - val_precision_m: 0.9977 - val_recall_m: 0.9972
```

Fig 7: Model Accuracy

# Result Analysis

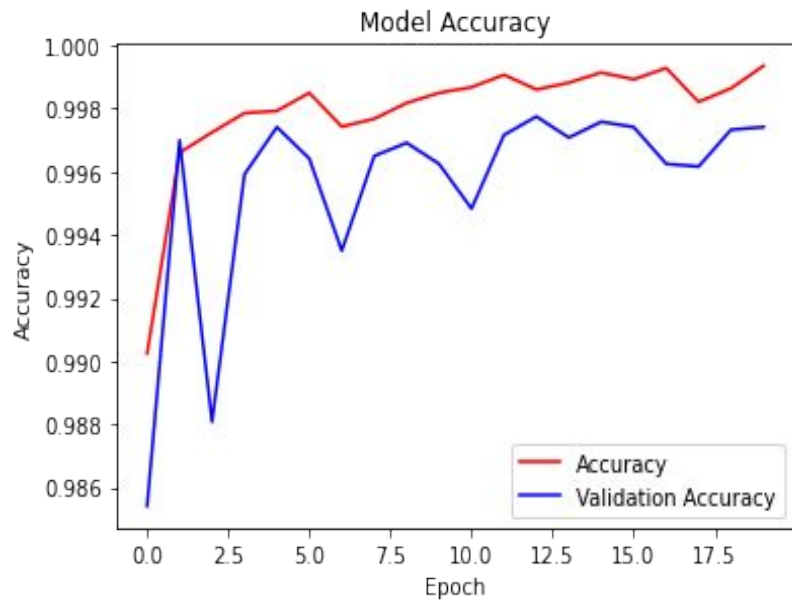


Fig 7: Accuracy vs Epoch Curve (lr=0.001)

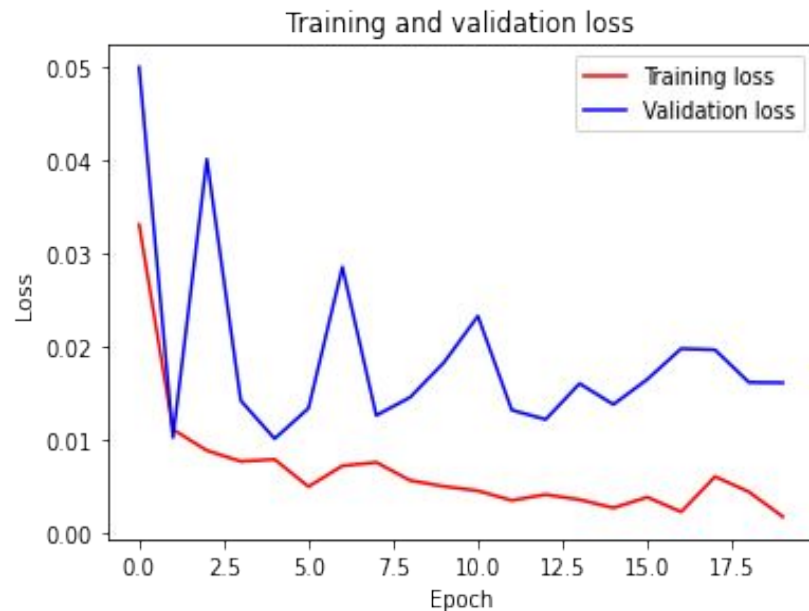


Fig 8: Loss vs Epoch Curve (lr=0.001)

# Result Analysis

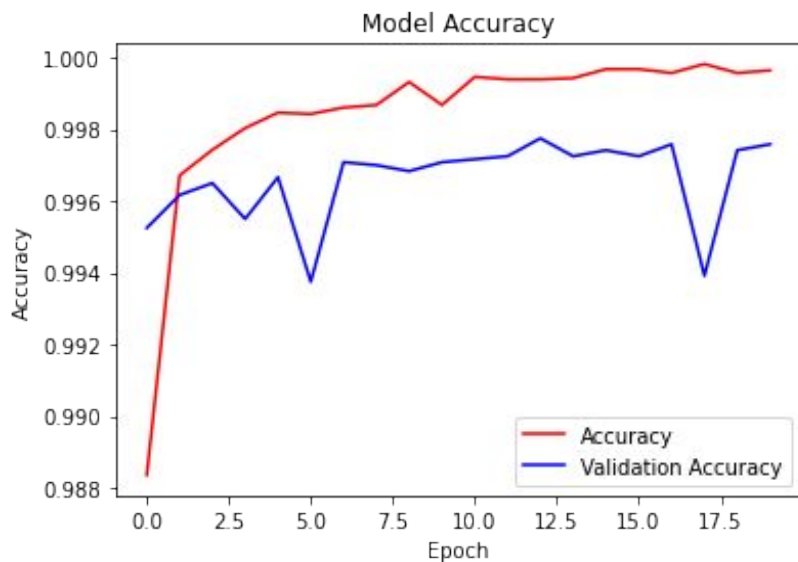


Fig 7: Accuracy vs Epoch Curve ( $lr=0.0001$ )

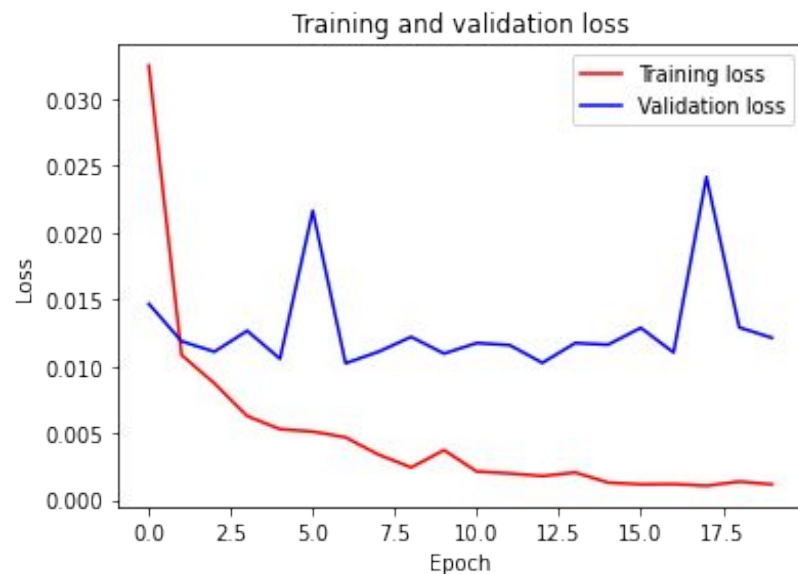


Fig 8: Loss vs Epoch Curve ( $lr=0.0001$ )

# Result Analysis

Performance of Different Methods:

Method	Accuracy	Recall	Precision	F1 Score
<b>VGG16 [2]</b>	0.998	0.999	0.998	0.998
<b>Inception V3 [2]</b>	0.997	0.997	0.998	0.997
<b>ResNet [2]</b>	0.975	0.945	0.994	0.968
<b>Proposed VGG16</b>	0.999	0.9993	0.999	0.999

Table 1: Performance of Different Methods



# Conclusion

1. The dataset was preprocessed before feeding the network
2. VGG16 was successfully implemented and validated
3. Performance analysis was done

## **Future Works:**

1. Extend the dataset using augmentation
2. Performance comparison of different neural network models

# References

- [1] <https://deeplearninganalytics.org/detection-of-surface-cracks-in-concrete-structures-using-deep-learning>.
- [2] Dung, C.V., 2019. Autonomous concrete crack detection using deep fully convolutional neural network. *Automation in Construction*, 99, pp.52-58.
- [3] Zhang, L., Yang, F., Zhang, Y.D. and Zhu, Y.J., 2016, September. Road crack detection using deep convolutional neural network. In *2016 IEEE international conference on image processing (ICIP)* (pp. 3708-3712). IEEE.
- [4] <https://medium.com/towards-artificial-intelligence/the-architecture-and-implementation-of-vgg-16-b050e5a5920b>
- [5] <https://towardsdatascience.com/adam-latest-trends-in-deep-learning-optimization-6be9a291375c>
- [6] <https://keras.io/api/preprocessing/image/>

THANK  
YOU