Identification of Cracks in Buildings with Deep Learning for Edge Devices

Akash R[1], Rakesh Teja P V [2], Khoushal [3]
UG Scholars,
School of Computer Science and Engineering,
Vellore Institute of Technology, Chennai Campus,
Tamil Nadu, India,

Abstract - **Modernization and urbanization have led to an exponential increase in number of high-rise buildings and structures such as bridges, dams, and factories. Defects such as cracks caused due to ageing and external factors such as environmental variations can severely affect the integrity of these structures and cause adverse consequences if left untreated. It is crucial to identify such cracks at the earliest. Identifying cracks manually in complex structures can be challenging. This project aims at providing a deep neural network-based solution to address this issue. To identify the presence of cracks, the project leverages the power of Convolutional Neural Network (CNN) models through transfer learning. A comparative study between four models namely InceptionV3, MobileNet, VGG-16 and ResNet50 is provided to identify the best performing model. Along with these classification models, a segmentation model namely YoloV5 is also presented which is used to detect cracks by drawing bounding boxes around them.**

I. INTRODUCTION

Growing population and technological advancements have led to an increase in big and complex infrastructures. Early crack detection is crucial because undetected cracks can grow over time, seeping deeper and deeper into the building. Leaving cracks undetected for too long could even lead to catastrophic downfall of parts or the entire building itself, resulting in huge losses. Manually scouring each nook and corner of buildings is not possible due to the complexity and size of buildings these days. It also requires a lot of resources and manpower. To add to that, there is always a possibility of human error. A crack may not be spotted as it may not be seen as significant enough, provided the challenging working conditions and external factors. Using automated algorithms will make the task much easier and more accurate. A drone being piloted across the buildings will make use of a camera affixed to it and collect data. Drones equipped with specialized sensors can even penetrate hard-to-reach areas, such as the undersides of bridges or the interiors of tunnels, providing comprehensive inspections without the need of manual intervention. But drones operate with finite power and processing constraints, making it crucial to optimize the algorithms used for real-time identification of cracks. Heavy computational tasks drain battery life rapidly, limiting the duration and coverage. There is a need for a lightweight, efficient model that ensures that the drone can operate for extended periods, maximizing its utility in large-scale inspections without the need for frequent recharging or battery swaps. Along with efficiency, accuracy plays a pivotal role in ensuring the effectiveness of crack detection systems integrated with drones. While computational efficiency enables real-time processing and extended flight times, accuracy is essential for reliable identification and classification of cracks. Achieving a delicate balance between efficiency and accuracy is necessary for optimizing the performance of these systems. This project aims at determining the best model which can offer an excellent balance between accuracy and computational efficiency. The video data from the drone is processed frame by frame through the model which then classifies crack frames and sends them to the user. The project achieves this by using convolutional neural networks to identify the cracks. The objective of the project is to automate the process of crack identification with the help of a trained model which has learned the features of crack images and identifies them with high accuracy. Convolutional Neural Networks are a class of deep neural networks that perform operations on images to obtain some specific pattern and features. It is passed through multiple layers where each take part in classifying the final result. They use a variety of filters to extract maximum information from every image. Backpropagation takes place across multiple layers which helps to assign more accurate weights to the model. Training through this network helps to identify the distinctive features of cracks and differentiate them from other items, and classify them properly. But training a new model with a new dataset from scratch may not be effective due to increased training time. The model may require more effort to learn effective representations and patterns, potentially leading to degraded performance. To overcome these difficulties, transfer learning can be used. Transfer learning is a technique where a model which was previously trained on other datasets is used to build a new model by providing our custom dataset. Since the model has pre existing knowledge, it can apply this knowledge for extracting features and making predictions. Using this on custom dataset speeds up the training process and improves the results. To identify crack images, CNN's which are pre trained are used and trained on crack image dataset through transfer learning. In this project, we have implemented crack detection models using five different architectures and provided results to highlight the best suitable model. The four architectures namely InceptionV3, MobileNet, VGG-16 and ResNet50 are all trained on ImageNet dataset. ImageNet consists of over 14 million images covering more than 20,000 categories or classes. Being trained on such vast dataset, the models have gained a very rich knowledge on feature extraction, which can be leveraged for our task by using transfer learning approach. To provide information regarding the location of cracks in images, a detection model namely YOLO V5 has also been implemented and the results of all the models have been provided as a comparative study.

II. LITERATURE REVIEW

The inception model, introduced by Christian Szegedy et al. [1], represents a significant advancement following the success of AlexNet. This architecture, known as GoogLeNet, was specifically designed by Google to address computational constraints while maintaining high performance, making it efficient and adaptable. The key feature of the model is its ability to achieve high performance with fewer parameters and lower computational costs. It is achieved by avoiding bottlenecks, maintaining high dimensions, reducing dimensions before spatial aggregation, and balancing the depth-width ratio of the network. By factorizing convolutions, particularly those of large sizes, the input size is maintained while reducing parameters and computational costs. Additionally, asymmetric convolution, which involves using a combination of 1xN and Nx1 convolutions, has proven to be more cost-effective. The inception model also introduces auxiliary classifiers, enhancing performance in the final stages of the network. Another innovation is the utilization of parallel blocks to reduce grid size, mitigating potential bottlenecks in the network.

Long D Nguyen et.al[2] have used Inception v3 along with two other CNN networks through transfer learning to classify microscopic images, where all the networks are pre trained on a very large dataset. The features are extracted from all three models and are concatenated together to form a large vector. This is sent further into two layers for classification, this is where the structure differs, by adding one more layer than normal, the network learns a lot more and is able to classify the microscopic image data. They also have deployed an algorithm that gives adaptive learning rate to maintain the convergence of the model. The loss function used is a combination of the crossentropy function and the loss function to achieve optimisation and stability.

Andrew G.Howard et.al [3] proposed the MobileNet model in their paper. MobileNet's architecture is designed in such a way that all the operations are not performed at once but split into two different operations. First a depthwise convolution that gives a single filter to each input and then a pointwise convolution which performs a 1*1 convolution to combine all the depthwise operations occurs. Computation is much easier due to this split. Since the model is very small, it is not as susceptible to incorrect classification as other models. Mobile Nets use two hyperparameters where one of them being the width multiplier and the other one being the resolution. Both are values between 0 and 1 and are used to thin down the model and to take care of input resolution. They are used to scale down the model to even smaller sizes when necessary.

Wang et al. [4] proposed Dense-MobileNet model which combines DenseNets and MobileNets to create a lightweight deep neural network with fewer parameters and improved classification accuracy. In this approach, the dense blocks, which are a key feature of DenseNets, are introduced into MobileNet models. The dense blocks consist of convolution layers with the same size of input feature maps, and dense connections are established within these blocks. These dense blocks allow the network to make full use of the output feature maps generated by previous convolution layers, generating a large number of feature maps with fewer convolution cores. By repeatedly using the features and the generating more feature maps with fewer convolution kernels, reduction in network parameters and computational complexity occurs, making the model particularly suitable for deployment on mobile devices with limited memory resources.

Tammina et al [5] proposed a model that uses VGG-16 as a feature extractor trained using transfer learning. The model is trained in a small dataset. To prevent the issue of overfitting which occur due to limited dataset, data augmentation techniques have been used to generate more data. The weights of all 5 convolutional blocks in the VGG-16 model were frozen, and the output was fed to a new classifier. Our project uses a very large dataset which provides the model rich features to learn from, thus enhancing the overall performance.

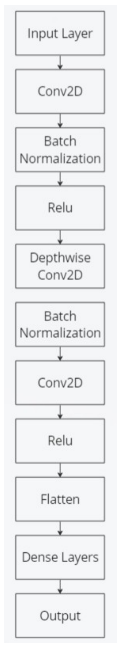
Xinglong et al. [6] proposed ResNet50-based approach for classifying surface defects in hot-rolled strip steel. It involves using a modified version of the ResNet50 architecture, incorporating attention mechanisms such as CBAM and FcaNet to improve the classification accuracy of surface defects. The approach aims to address the impact of surface quality on the final product in industries such as automotive manufacturing, chemical, and home appliance industries. By enhancing the ResNet50 model with attention mechanisms, the approach seeks to improve the classification ability of the algorithm and ultimately contribute to the quality control of hot-rolled strip steel products.

Jung et al. [7] proposed a improved version of YOLOv5. In this model, 6x6 Conv2d layer is replaced with Focus layer, the C3 blocks are replaced with BottleneckCSP and the SPP layer with the SPPF layer. These replacements result in more than double the computational speed, making the proposed model more efficient and faster. It achieves a higher mean average precision (mAP) and shows better convergence speed in the loss function during training.

III. PROPOSED METHODOLGY

We have proposed two different architectures. One architecture can be used for classification cracks and the other can be used for detecting them by drawing bounding boxes. When a crack is identified in the image, it is sent to the user. The user can use any of the two models as per the requirement. When image is classified as crack image, the user should still see the image and identify the crack manually. Classification provides higher accuracy than detection models because they only identify the majority class rather than predicting bounding box and locating them than detection models. If a user wishes to automate the detection process also, then they YOLO model can be used. But MobileNet model is recommended.

1. MobileNet Architecture



Input Layer: This layer receives the input image. In MobileNet, the input image size is typically 150x150 pixels.

Conv2D: This is the first convolutional layer. It applies filters extracts features from the input image using learned filters and stores them in feature maps. Convolution is an important operation in CNNs as it's responsible for learning spatial features from the input data and reduces spatial dimension after extracting features
Batch Normalization: This layer normalizes the outputs of the previous layer. This helps with training speed and stability.

ReLU: This is the rectified linear unit activation layer. It introduces non-linearity into the network. ReLU activation functions by setting all negative outputs to zero.

Depthwise conv: This is the core building block of MobileNet. Depthwise convolution filters each input channel independently using lightweight filters. If there are 3 channels like RGB and 32 filters are used, then these filters will individually operate on each of the channels and extract features.

Pointwise conv: It uses 1x1 filters to combine the outputs from the depthwise convolution layer. This approach significantly reduces the number of computations compared to a standard convolution layer, making MobileNet more efficient for mobile and edge devices.

Batch Normalization: Another batch normalization layer is applied to normalize the outputs of previous layers.

ReLU: Another ReLU activation layer is used.
Flatten: This layer flattens the output from the previous layer into a one-dimensional vector. This is necessary for feeding the data into the fully-connected layers.

Dense Layers: These are fully-connected layers that are responsible for image classification. Since each neuron is connected to every other neuron, it can capture complex patterns. It also maps the output to desired number of class labels.

Output Layer: This layer outputs the final predictions of the network. For image multi-class classification tasks, it would contain a softmax function that outputs a probability distribution over all classes. If it is binary classification, then it would use sigmoid function.

2. YOLOV5s Architecture

To implement an algorithm in drones, it is necessary that it is computationally efficient and requires less resources, but at the same time should offer high accuracy. For this purpose, YOLOV5s (You Only Look Once – Version 5 (Small)) model has been used. The 5s variant offers a phenomenal balance between speed and accuracy due to its computational efficiency. The YOLOV5s model uses CSPDarknet53 as backbone. It is a modified version of the Darknet53 network with Cross-Stage Partial connections (CSP) for efficient processing. The backbone serves as the feature extractor. To make the model even more efficient, we have modified the pre-existing backbone by introducing Focus and BottleneckCSP layers.

Focus layer: It improves feature representation, especially for smaller input image. It takes the input image and divides it into four equal rectangular sections along the width and height dimensions. These four sections are then stacked together along the channel dimension, quadrupling the number of channels in the feature map. This allows the network to learn spatial relationships between features in different parts of the image earlier in the processing pipeline.

BottleneckCSP: It is a modified version of C3 block. It helps in achieving a balance between feature richness and computational efficiency. The C3 block comprises two parallel convolutional branches with 3x3 filters. In the first branch, the input feature map is split into half and one half of the features is passed through a bottleneck layer (1x1 convolution) for feature compression. The second branch processes the other half directly. The outputs from both convolutional blocks are then concatenated, resulting in a feature map with the original number of channels but capturing information at different resolutions. In BottleneckCSP block, a 1x1 bottleneck convolutional layer is initially present before the feature map splitting. This layer has significantly fewer neurons compared to the layers before and after it. It creates a compressed representation of the input data by projecting it into a lower-dimensional space. This reduces the channel dimensionality of the input feature map, resulting in significant computational efficiency. While reducing dimensions, the network is forced to learn the most important and representative features of the input data. This compression acts like a form of regularization, potentially helping to prevent overfitting. The compressed features are then split in half and passed into two branches similar to the standard C3 structure. Using bottleneck convolution initially before splitting features help in decreasing the number of parameters and computations compared to a standard C3 block. The BottleneckCSP layer has also been used in the head block of the architecture. The CSP layers in Fig 1. refers to the BottleneckCSP layers. The number of classes is modified from 80 to 1 in the architecture. YOLO is an object detection model and trained to detect over 80 different objects. In this project, the model is trained only to detect the cracks from images. Thus, the class number is changed to 1.

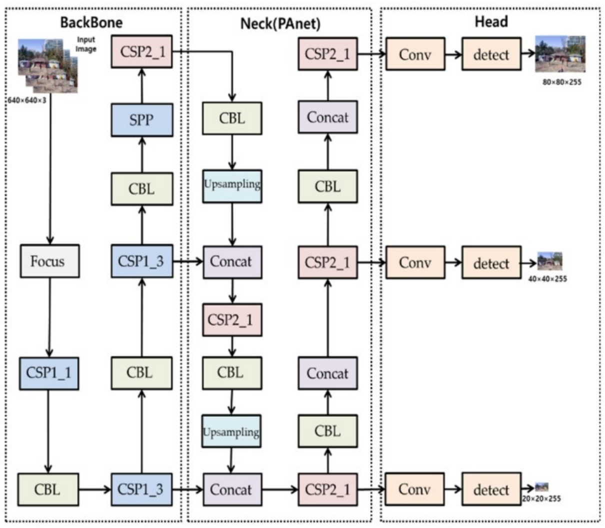


Fig 1. Proposed architecture of YOLOV5s model [7]

BLOCK DIAGRAM

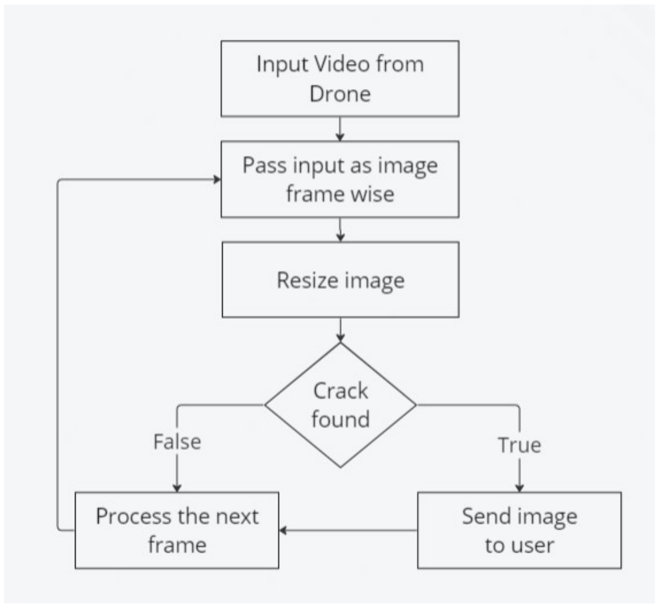


Fig 2. Block Diagram of the proposed crack detection system

The input video is initially received from drone. The video is then split into frames and each frame is passed as input into the model. The model then makes predictions. If a crack is identified in that particular frame, it is sent to the user as image. This helps the user to identify and locate the place where the crack has been found. If no crack is predicted in the frame, the model proceeds to process the next frame.

IV. DATASET DESCRIPTION

For the classification models, the dataset used contains a total of 40,000 images where 20,000 images belong to the positive class indicating presence of crack in surface and the remaining 20,000 images belong to the negative class indicating absence of cracks. Out of 40,000 images, 22,400 images were taken for the training set, 5600 for validation and 12,000 images for testing the performance of the model. For the YOLOV5s model, the crack dataset contains 3717 training images, 402 validation images and 324 test images along with the labels containing segmentation data as text file for each image. All images used for classification models are of size 227 × 227 pixels resolution and 416 x 416 pixels for yoloV5 model. Both the image datasets are in JPG file format.

V. PRE-PROCESSING STEPS

The images are initially loaded and resized according to the requirement of the architectures. Image augmentation techniques have not been used because since the dataset contains 40000 images which is very vast, the model can learn enough features from the images without the need of augmentation. Augmentation techniques are generally used to increase the number of dataset images so the model can learn to extract features with high accuracy. But in our project, the vast dataset combined with transfer learning has already helped the model to produce the best results.

VI. EXPERIMENTAL RESULTS

The four classification models were evaluated on the test data and the following confusion matrices were obtained. A confusion matrix summarizes the predictions of the model against the actual outcomes, displaying TP, TN, FN and FP. Confusion matrices help in deducing valuable insights into a model's performance and understanding the effectiveness of the model in classification tasks. In Table 1, the confusion matrix data of four classification models is given to highlight the performance measures of the model.

TABLE I. CONFUSION MATRIX OF FOUR MODES

Model	***TP***	***FN***	***FP***	***TN***
InceptionV3	6051	18	14	5917
MobileNet	6041	28	3	5928
VGG	6038	31	79	5852
Resnet50	6010	59	609	5322

From the confusion matrix data, both InceptionV3 and MobileNet have seemed to perform really well. They provide high number of true positives, and lower false positives and false negatives. MobileNet has the least number of false positives whereas ResNet50 has an alarming number of false classifications. On the other hand, VGG-16 has provided satisfactory results. With the help of these four parameters TP, TN, FP and FN, the Precision, Recall and F1 Score of the model can be calculated.

Precision (P): Measures the accuracy of positive predictions made by the model. It is the ratio of correctly predicted positives (TP) to the total predicted positives (TP+FP). Precision provides information regarding the reliability of positive predictions.

$$Precision = \frac{TP}{TP + FP}$$

Recall (R): Measures the ability of the model to capture all the positive instances. It is the ratio of correctly predicted positives (TP) to the total actual positives (TP+FN).

$$Recall = \frac{TP}{TP + FN}$$

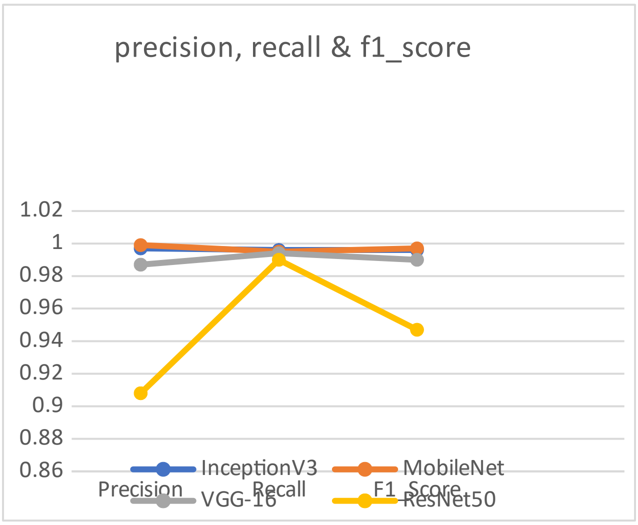
F1 score: It is the harmonic mean of precision and recall. It offers a balanced overview about the model, particularly valuable in scenarios with imbalanced class distribution, by considering both the precision and recall of a model.

$$F1\,Score = \frac{2 * Precision * Recall}{Precision + Recall}$$

TABLE II. PRECISION, RECALL, F1_SCORE OF ALL MODELS

Model	***Precision***	***Recall***	***F1 Score***
InceptionV3	0.997	0.996	0.996
MobileNet	0.999	0.995	0.997
VGG	0.987	0.994	0.990
Resnet50	0.908	0.990	0.947

MobileNet has the highest value for precision, indicating its reliability in positive predictions. Inception has also given significant results but was still outperformed by MobileNet.

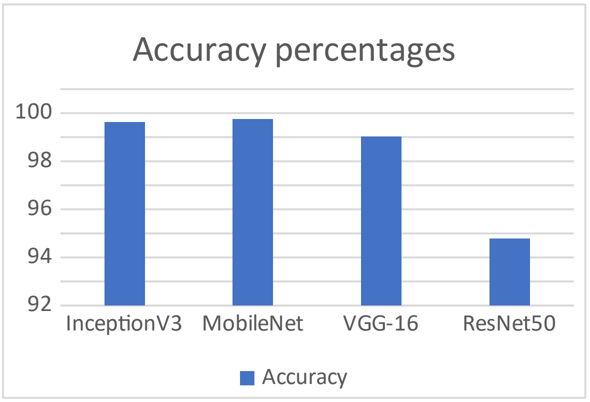


To determine how accurate the model is in identifying cracks, accuracy percentage is calculated.

$$Accuracy\ Percentage = F1\,Score * 100$$

TABLE III. ACCURACY VALUES OF FOUR MODES

Model	***Accuracy***
InceptionV3	99.62
MobileNet	99.74
VGG-16	99.02
ResNet50	94.78



MobileNet has again outperformed the other models by providing the highest accuracy. From all the obtained results, MobileNet has turned out to be the most effective architecture. Due to its depthwise seperable convolutions, it is a lightweight model and thus the computation cost is reduced. Its architecture is designed in such a way that the runtimes are faster than traditional models. This results in faster and more reliable predictions. On the other hand, ResNet50 faces the issue of vanishing gradients. Even though residual connections help address the vanishing gradient problem, very deep ResNet architectures may still face some training difficulties and degrades output.

RESULTS OF YOLO MODEL

The results of the YOLO model have been presented separately because YOLO is a detection model and was trained with a different dataset. The results of YOLOV5s model are relatively low compared to the classification models. This is because it must not only classify crack images but also precisely locate and draw bounding boxes around the cracks. Additional factors such as object loss and box loss are also considered which affect the final result values.

TABLE IV. PERFORMANCE OF YOLOV5S MODEL

Metrics	***Values***
Precision	0.852
Recall	0.731
F1 Score	0.786
Accuracy	78.68

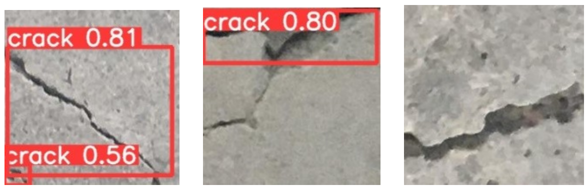


Fig 2 Predictions of YOLOV5s model

I. CONCLUSION

This project underscores the importance of leveraging Convolutional Neural Networks (CNNs), for crack detection in high-rise buildings and critical infrastructure. Through transfer learning, we evaluated four popular CNN models, namely InceptionV3, VGG-16, ResNet50, and MobileNet, along with the YOLOv5 segmentation model. Our findings demonstrate that MobileNet emerges as the most suitable choice, offering a balance between high accuracy and computational efficiency. Its lightweight design makes it well-suited for deployment in resource-constrained environments, while still delivering robust performance in crack detection tasks. VGG16's depth and large number of parameters make it a heavy model and is not suited for applications with memory or speed constraints, as its large size leads to slow inference times and high memory usage. Inception provides good results but is again a heavier model compared to MobileNet. It is crucial for the algorithm to be light weight and efficient to be deployed with edge devices. MobileNet has emerged as the best model for this purpose.

VII. REFERENCES

[1] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathan Shlens, Zbigniew Wojna -"Rethinking the Inception Architecture for Computer Vision"

[2] Long D. Nguyen, Dongyun Lin, Zhiping Lin, Jiuwen Cao-"Deep CNNs for microscopic image classification by exploiting transfer learning and feature concatenation"

[3] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, Hartwig Adam-“MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications”

[4] Wang, Wei, et al. "A novel image classification approach via dense-MobileNet models." *Mobile Information Systems* 2020 (2020).

[5] Tammina, Srikanth. "Transfer learning using vgg-16 with deep convolutional neural network for classifying images." *International Journal of Scientific and Research Publications (IJSRP)* 9.10 (2019): 143-150.

[6] Feng, Xinglong, Xianwen Gao, and Ling Luo. "A ResNet50-based method for classifying surface defects in hot-rolled strip steel." Mathematics 9.19 (2021): 2359.

[7] Jung, Hyun-Ki, and Gi-Sang Choi. "Improved yolov5: Efficient object detection using drone images under various conditions." Applied Sciences 12.14 (2022): 7255.

[8] Golding, Vaughn Peter, et al. "Crack detection in concrete structures using deep learning." Sustainability 14.13 (2022): 8117.