Indian Journal of Information Sources and Services
ISSN: 2231-6094 (P) Vol.14, No.1, 2024, pp.93-100

DOI: https://doi.org/10.51983/ijiss-2024.14.1.4003

Advancing Crack Detection Using Deep Learning Solutions for Automated Inspection of Metallic Surfaces

Snehal Rathi[1], Omkar Mirajkar[2], Laukik Deshmukh[3], Shubhangi Shukla[4] and Lokesh Dangare[5]
[1]Assistant Professor, [2,3,4&5]UG Student,
Department of Computer Engineering, Vishwakarma Institute of Information Technology, Pune, Maharashtra, India
E-mail: snehal.rathi@viit.ac.in, omkar.22010639@viit.ac.in, laukik.22011070@viit.ac.in,
shubhangi.22010512@viit.ac.in, lokesh.22010444@viit.ac.in
(Received 12 January 2024; Revised 29 February 2024, Accepted 20 March 2024; Available online 30 March 2024)

***Abstract* - The deep Convolutional Neural Network (CNN) architecture used in this research study provides a proof of concept for crack detection on the metallic surface of a hex nut. The goal is to create an automated receiving inspection process to supplement human inspections conducted on-site. Conventional image processing techniques (IPTs) have been extensively used for mechanical infrastructure fault detection. These techniques focus on image modification to extract typical features, such as surface fractures in materials like steel and concrete. However, obstacles presented by a variety of real-world variables, such as changes in lighting and shadows, make it difficult to use IPTs. Our suggested vision-based method employs a deep learning CNN to overcome these difficulties, eliminating the need to explicitly compare fault features. CNNs are more resilient to shifting real-world situations than IPTs since they are naturally trained to identify characteristics in images. Following training on a dataset of 1081 images with dimensions of 256 x 256 pixels, the VGG16 CNN architecture achieved an impressive accuracy of around 94.17%. Additional CNN architectures, including ResNet, MobileNet, AlexNet, and LeNet-5, are employed to assess and compare fault detection accuracies in order to select the most appropriate architecture for the model. To evaluate the robustness and flexibility of the suggested method in various situations, we conducted tests with 206 images from an alternative structure that was not part of the training dataset. These images depicted a range of circumstances, such as intense light patches and tiny fissures. The outcomes showed that our proposed method outperforms current approaches, highlighting its usefulness in practical situations involving the identification of metallic defects.**
Keywords: **Convolutional Neural Network, Crack Detection, Automated Receiving Inspection, Image Processing Techniques, Vision-Based Method, Deep Learning, VGG16 CNN, ResNet, MobileNet, AlexNet, LeNet-5**

I. INTRODUCTION

The structural integrity of numerous engineering systems, including equipment, structures, and bridges, hinges significantly on the condition of nuts. Timely detection of cracks in nuts is imperative to avert catastrophic failures and uphold the safety of these structures (Johnston, 2019). In recent years, Convolutional Neural Networks (CNN) have emerged as highly effective tools for image-based defect identification, owing to advancements in computer vision and deep learning (Le Cun *et al.,* 2015). This study delves into the application of CNN for crack detection in nuts, with the primary objective being the identification and localization of cracks, accompanied by an assessment of the strengths and weaknesses of different approaches.

Deep learning models are preferred for crack detection due to their ability to autonomously extract hierarchical features from data, obviating the need for explicit feature engineering (Bengio *et al.,* 2013). Unlike traditional methods that often rely on handcrafted features, CNN can learn relevant features directly from the data through convolutional layers, potentially offering more accurate and robust crack detection in nuts.

The identification of strengths and weaknesses inherent in each model is pivotal for guiding the selection of an appropriate approach tailored to specific application requirements. Moreover, this study contributes to the expanding body of research aimed at enhancing efficiency and reliability in defect detection (Smith *et al.,* 2020).

A comprehensive overview of relevant work in the fields of defect detection and deep learning applications precedes the methodology section. The methodology entails detailing the dataset used, model architectures, and training procedures. Subsequently, results will be presented and analyzed. Finally, the conclusion will summarize major findings and propose directions for further investigation.

II. SUMMARY OF THE PROPOSED MODEL

The proposed model encompasses a comprehensive workflow aimed at leveraging CNN for crack detection in nuts. High-resolution images of metallic surfaces on nuts, capturing various image fluctuations such as illumination and shadows, are obtained using a camera. These images are then utilized to train a CNN classifier, with cracks defined as imperfections visible to the naked eye in the pictures.

The dataset comprises 1287 raw images, including 1081 images for training and validation and 206 images for testing, each with varying pixel resolutions. To establish a database, the 1081 images are manually re-sized to small dimensions (256 x 256 pixels), from which cropped images are randomly

selected to construct training and validation sets. The CNN classifier is subsequently trained using the prepared training image set to distinguish between fractured and intact metallic surface images.

III. LITERATURE REVIEW

The focus of the literature review is on employing CNN for defect identification in industrial contexts, particularly about metal nuts. It addresses issues with conventional techniques and investigates the effectiveness of CNN in feature extraction and pattern recognition. Our understanding lays the path for further investigation into issues related to defect detection and innovations, particularly about metal nuts (Sauter *et al.,* 2021).

Convolutional Neural Networks (CNN) are robust computer vision techniques that facilitate the vision-based defect crack detection process. Our focus is on the latest advancements in CNN-based crack identification in varied lighting and shadow conditions. To enhance computational capacity and identify cracks faster than human labour, we evaluate manual procedures, image processing techniques (IPTs), and machine learning approaches (Cha *et al.,* 2017).

Deep learning contributes by utilizing a pre-trained model to extract the features and classify the defects in steel and metal. Utilizing LeNet, AlexNet, and VGG16, we were able to identify the flaws with progressively higher accuracy. VGG16 models have achieved up to 93% accuracy, and this accuracy can be increased by using more epochs. The lack of data in this case presents an over-fitting problem. Combining this with currently available technologies can assist us in determining the kinds of problems and the steps that must be taken to mitigate damage (Piwal *et al.,* 2023).

ResNet 152 deep residual neural network architecture is the classifier that we may use to identify abrasions, scratches, and scrapes on metal surfaces. Based on test data, it provides us with a classification accuracy of 97.14%. During our training process, we have determined which augmentation circumstances have the biggest impact on enhancing the model's accuracy metrics. Research has also been done on damage peculiarities that make it difficult to identify them. As Python is a quick programming language, we used the "Keras" and "TensorFlow" packages to construct our recommended solution. Enhancing the method for examining metal surface operations and adjustment parameters can benefit from the achieved result (Konovalenko *et al.,* 2021).

Through the fine-tuning of the AlexNet Model, we presented in the research a CNN-based crack detection technique. In this work, Initially, we employed the 4160x3120 pixel raw images, cropped into 256x256 pixel images, to construct the training and validation set then the CNN model was trained using the refined AlexNet model having an accuracy of 98.67% (Shengyuan *et al.,* 2017).

IV. METHODOLOGY

The general architecture, the layers employed in the investigation, and the histories of each layer are all explained in this part. Multiple layers, including input, convolution, pooling, activation, and output layers, can be used to form the overall CNN architecture. Convolution and pooling operations are carried out in the convolution and pooling layers. A neural network with a multi-layered design is called a deep neural network. Depending on the intended uses, additional auxiliary layers like batch normalization (BN) and dropout layers may be incorporated into the aforementioned layers.

A. Convolution Layer

An essential procedure that underpins a neural network's capacity to automatically identify and extract hierarchical characteristics from input data is the convolution layer operation, most especially in Convolutional Neural Networks (CNN). Sliding tiny filters, or kernels, across the input data and carrying out a convolution operation at each place constitutes the fundamental part of this operation. The filter calculates the dot product between its weights and the associated values in the input at each step by scanning a local receptive field of the input.

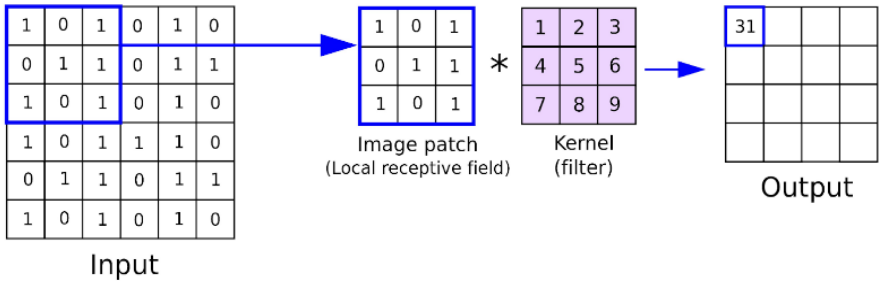

Fig. 1 Example of CNN having Convolutional Layer

The result is a feature map that highlights relevant patterns, such as edges, textures, or more complex structures, depending on the task at hand. The convolution operation inherently captures spatial hierarchies, enabling the network to discern local features in lower layers and progressively combine them into more abstract and complex representations in higher layers. This process allows CNN to automatically learn and adapt to the intricate patterns inherent in the input data, making them particularly effective in image-related tasks such as crack detection in metallic raw

materials. The convolution layer operation, coupled with weight sharing and hierarchical learning, forms the backbone of CNN, contributing to their success in a variety of machine-learning applications.

B. Pooling Layers

In the architecture of Convolutional Neural Networks (CNN), pooling layers are critical because they reduce the spatial dimensions of feature maps, which improves computational efficiency and the network's capacity to concentrate on important information. The pooling operation involves systematically down-sampling the input data within local regions defined by small windows. The most popular kind of pooling is called max pooling, which successfully highlights the most salient features by retaining the greatest value inside each frame.

Alternatively, average pooling calculates the average value in each window. The pooling operation introduces translation in-variance, allowing the network to detect features regardless of their exact position in the input space. By reducing the spatial resolution, pooling layers also contribute to parameter reduction and mitigation of over-fitting, aiding the network's generalization capabilities. However, careful consideration is needed to balance down-sampling and information preservation, ensuring that crucial features are retained for accurate and meaningful representation in subsequent layers of the CNN architecture.

C. Activation Layer

An essential part of neural network architectures, such as convolutional neural networks (CNN), is the Rectified Linear Unit (ReLU) activation function. Its function is to add non-linearity to the network by introducing all negative values to zero and permitting positive values to flow through unaffected. Mathematically expressed as f(x)=max(0,x),

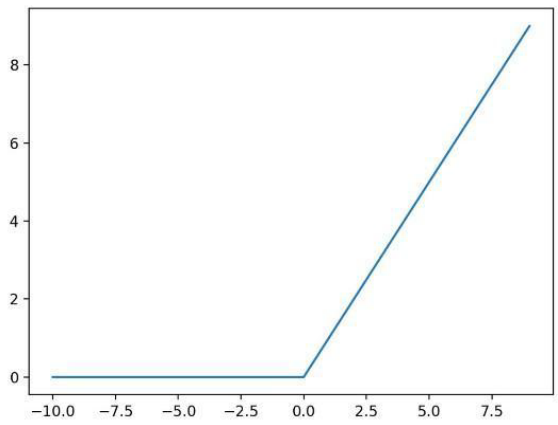

Fig. 2 ReLU Activation Layer

The ReLU operation efficiently addresses the vanishing gradient problem, promoting faster and more effective convergence during training. By introducing non-linearity, ReLU enables neural networks to model complex relationships in data, contributing to their capacity to learn and represent intricate patterns. The simplicity and computational efficiency of the ReLU activation have made it a popular choice in deep learning models, enhancing their ability to capture and propagate relevant information through the network layers. This activation function has played a pivotal role in the success of various applications, from image recognition to natural language processing, contributing to the advancement of deep learning across diverse domains.

D. Auxiliary Layers

Over-fitting has long been an issue in machine learning. This issue occurs when a network successfully classifies a training set of data, but the results of testing and validation remain unsatisfactory. To address this issue, dropout layers are used. Because of intricate co-adaptations, training a network with a large number of neurons frequently leads to over-fitting. Dropout's basic concept is to arbitrarily break connections between neurons in linked layers at a specific dropout rate. Reducing these adaptations enables a network to generalize training instances considerably more effectively.

E. Sigmoid Function

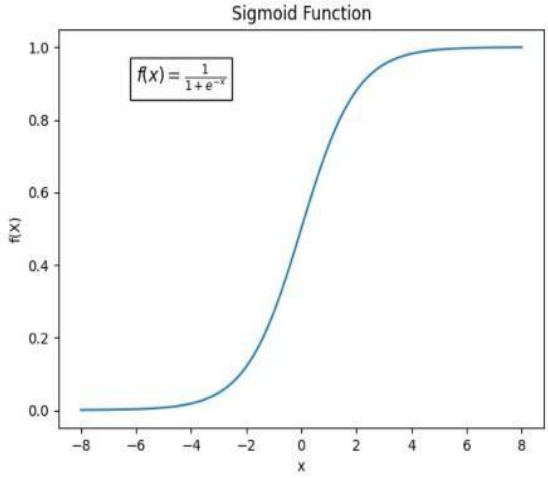

Fig. 3 Sigmoid Function

The sigmoid function, often denoted as σ(x), is a popular activation function used in neural networks, logistic regression, and binary classification tasks. It maps any real-valued number to the range [0, 1], making it particularly useful for producing probabilities. The sigmoid function's mathematical expression is as follows:

$$f(x) = \sigma(x) = \frac{1}{1 + e^{-x}}$$

Fig. 4 Sigmoid Function Mathematically

The mathematical constant Euler's number, represented by the letter "e" in this case, is roughly equivalent to 2.71828. The input "x" is effectively squashed to a probability-like scale by the sigmoid functions, which accept it as input and return a value between 0 and 1. When "x" is large, the

sigmoid output approaches 1, and when "x" is significantly negative, the output tends toward 0.

The sigmoid function is advantageous in scenarios where the goal is to model binary outcomes or where the outputs need to be interpreted as probabilities. Its smooth, differentiable nature is conducive to efficient gradient-based optimization during the training of neural networks. Despite its common use, the sigmoid function is not without limitations, such as susceptibility to vanishing gradient problems in deep networks. However, in binary classification tasks and scenarios demanding interpretable outputs, the sigmoid function remains a valuable tool in the arsenal of activation functions.

V. DEVELOPING A CLASSIFIER TO IDENTIFY SURFACE CRACKS

This section explains the factors taken into account when creating the database and underlying hyper-parameters that were set for CNN training. It takes time to configure and select appropriate hyper-parameters (such as learning rates and regularization parameters), and there are no precise recommendations for those parameter optimizations. As a result, using the validation set errors as a reference, it is required to experiment to find the optimal network architecture for this concrete crack detection. This article's tasks are all completed on a workstation equipped with GPUs (GPU: NVIDIA® GeForce RTXTM 3050, RAM: 16GB, and CPU: Intel® CoreTM i5).

A. Data Generation

To train and test the performance of the model, a total of 1081 raw images with various pixel resolutions are used. The images were captured with a hand-held, high-resolution 64 MP camera from mechanical hardware prototypes of hex-faced nuts. The objects were between 2.0 and 3.0 inches away on average. However, some test shots were taken at less than 1.0 inches, and the illumination in each shot varies significantly. 1081 of the 1287 raw images are used in training procedures, while 206 images are used in testing procedures. After labelling every image as intact or cracked, the 1287 raw images are cropped into smaller images with 256 x 256 pixel dimensions to construct the database for training and testing. As a result, there are 1287 images are there in the database. To create training sets, randomly selected images are taken from the database.

The network that was trained on small images allows the scanning of any image that is larger than the intended size, which is why the relatively small cropping size was chosen. However, the network might capture any elongated features, like scratches, if smaller images than the ones chosen here are used. Additionally, marking images as damaged or undamaged is more challenging for smaller images. The produced database contains a large range of image variations for a robust damage classifier. There are fractures on four of the picture space boundaries in several of the cropped images. These kinds of images are strictly prohibited for the reasons listed below. First, the size of the input images decreases as they pass through the CNN, indicating that edges containing fractures are less likely than those without to be picked up by a network during training. Second, it is impossible to determine if these fracture features are genuine cracks or not, which may result in incorrect annotations in the training dataset.

Lastly, the barely discernible crack features make it impossible to validate whether the predicted class is true-positive or false-positive, even if a trained model can classify such images.

Fig. 5 Bad metallic surface

Fig. 6 Good metallic surface

B. Data Augmentation

We used a data augmentation technique to create variations of an image that was taken from a piece of hardware. In the fields of computer vision and machine learning, data augmentation is a crucial technique that aims to improve model performance by artificially increasing the diversity of training datasets. This method involves applying various transformations to existing data, creating new instances with modified features while retaining the essential characteristics of the original samples. Common augmentation techniques include rotations, flips, translations, and changes in lighting conditions. The primary goal of data augmentation is to equip models with the ability to generalize well to unseen variations and improve robustness. By exposing the model to a broader range of scenarios, data augmentation mitigates over-fitting and aids in capturing the inherent variability present in real-world data. This strategy is particularly valuable when working with limited datasets, as it effectively amplifies the amount of training information available. Employing data augmentation adheres to ethical research practices, as it contributes to model fairness, transparency, and accuracy without introducing biases, ensuring the generation of reliable and effective machine learning models.

C. Hyper Parameters

Using the SGD technique with a tiny batch size of 16 images out of 1081, the described network is trained. The logarithmically decreasing learning rates are utilized because tiny and decreasing learning rates are advised (Wilson and Martinez, 2001). Momentum and weight decay parameters

are assigned by 0.9 and 0.0001, respectively. Our base model, a multilayered pretrained powerful model that can categorize images into many categories based on input, is based on the VGG16 architecture.

```
In [10]: from keras.preprocessing.image import ImageDataGenerator

In [ ]: datagen = ImageDataGenerator(
            rotation_range=30,
            width_shift_range=0.1,
            height_shift_range=0.1,
            rescale=1./255,
            shear_range=0.1,
            zoom_range=0.1,
            horizontal_flip=True,
            fill_mode='nearest')
```

Fig. 7 Code Snippets of Data Augmentation Conditioning

D. Architecture

Convolutional Neural Network (CNN) architecture is a novel paradigm in deep learning that is especially suited for tasks that require grid-like data, such as computer vision and image recognition. The hallmark of CNN is their hierarchical structure, characterized by layers that progressively learn hierarchical features from input data. Usually, the architecture is made up of fully connected, pooling, and convolutional layers. Convolutional layers use learnable filters to scan input data, capturing spatial hierarchies and local patterns crucial for understanding complex features. Pooling layers reduce spatial dimensions, focusing on the most salient information. High-level features are integrated by fully connected layers to produce final forecasts. We have compared the performances of multiple pre-trained transfer learning model architectures including VGG16 Net, Res152 Net, MobileNet, AlexNet, and LeNet out of which VGG Net and ResNet performed spontaneously and were able to classify the images with higher accuracy as compared to other architectures.

In our model architecture, we have built a simple model architecture with VGG and Res Net Models. The initial convo layers pairs of the VGG model include a filter of size 64 x 64 kernel size of 3x3 with an activation function ReLU. This layer intakes an input pre-processed image of size 256 x 256 x3 RGB image and performs convolution operations on it. The following pair of convo layers have filters of size 128 x128 and 256 x 256 with the same kernel size and ReLU activation function. The final 2 blocks of VGG16 Net include 3 convo layers in each block with a filter size of 512 x 512. Model architecture also includes Max pooling layers after each convo layer of size 2 x 2 to reduce the spatial dimensions of the output image from each layer. Following consecutive convo and pooling layers, the output image is flattened in a one-dimensional vector and given input to the Dense network with 3 layers of densely connected neurons. The activation Function for dense layers is the same as ReLU except for the last layer where we have used Sigmoid activation Function.

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[(None, 256, 256, 3)]	0
block1_conv1 (Conv2D)	(None, 256, 256, 64)	1792
block1_conv2 (Conv2D)	(None, 256, 256, 64)	36928
block1_pool (MaxPooling2D)	(None, 128, 128, 64)	0
block2_conv1 (Conv2D)	(None, 128, 128, 128)	73856
block2_conv2 (Conv2D)	(None, 128, 128, 128)	147584
block2_pool (MaxPooling2D)	(None, 64, 64, 128)	0
block3_conv1 (Conv2D)	(None, 64, 64, 256)	295168
block3_conv2 (Conv2D)	(None, 64, 64, 256)	590080
block3_conv3 (Conv2D)	(None, 64, 64, 256)	590080
block3_pool (MaxPooling2D)	(None, 32, 32, 256)	0
block4_conv1 (Conv2D)	(None, 32, 32, 512)	1180160
block4_conv2 (Conv2D)	(None, 32, 32, 512)	2359808
block4_conv3 (Conv2D)	(None, 32, 32, 512)	2359808
block4_pool (MaxPooling2D)	(None, 16, 16, 512)	0
block5_conv1 (Conv2D)	(None, 16, 16, 512)	2359808
block5_conv2 (Conv2D)	(None, 16, 16, 512)	2359808
block5_conv3 (Conv2D)	(None, 16, 16, 512)	2359808
block5_pool (MaxPooling2D)	(None, 8, 8, 512)	0

Fig. 8 VGG 16 Net Architecture

Similarly, the input to ResNet-152 is typically an RGB image with a fixed size, commonly re-sized to 256x256 pixels. The image is passed through the network for feature extraction and classification. The input image is passed through the initial convolutional layer, which performs a convolution operation with a large filter size (typically 7x7) to extract basic features from the input image. This layer is followed by batch normalization and ReLU activation to normalize the activation's and introduce non-linearity into the network.ResNet-152 consists of several stages, each containing multiple residual blocks. These stages gradually reduce the spatial dimensions of the feature maps while increasing the number of channels.

Each stage typically performs down-sampling (reducing spatial dimensions) either through convolutional layers with strides or through max-pooling operations. Within each stage, the residual blocks are stacked. ResNet-152 has 4

stages in total. Each residual block contains multiple convolutional layers along with skip connections. The basic building block of ResNet is the "Residual Block." In ResNet-152, the primary type of residual block used is the Bottleneck Block. Bottleneck Block consists of three convolutional layers: 1x1 Convolution: It's used to reduce the number of channels (dimensionality reduction). 3x3 Convolution: This layer performs the main feature extraction. 1x1 Convolution: This layer is used to increase the number of channels. Each convolutional layer is followed by batch normalization and ReLU activation.

The original input to the block is also passed through a shortcut connection (skip connection) to the output. The skip connection enables the gradient to bypass the convolutional layers, alleviating the vanishing gradient problem and allowing for easier training of very deep networks. After the final stage of residual blocks, global average pooling is applied to aggregate spatial information across the feature maps.

Global average pooling reduces each feature map to a single value by averaging all its elements, resulting in a fixed-size feature vector regardless of the input image size. The output of global average pooling is then passed through a fully connected layer. This layer serves as the final classifier and typically consists of a dense layer followed by a softmax activation function, which generates the final class probabilities. The output layer produces the final predictions, providing the probability distribution over the different classes in the classification task.

E. Training and Testing Results

The CNN model with an architecture is trained on an image set of batch size 16. Each batch includes a set of 16 pre-processed images of size 256 x 256 x 3. These 68 batches of images including in total of 1081 images are provided as training data for the CNN model. This dataset includes 1:1 ratio variable picture data of metallic surfaces for both surface crack-free and cracked surfaces. To prevent bias in the model's predictions, the equal ratio must be used during model training.

The model is trained on input images for 5 epochs and the weights of neurons are adjusted using back-propagation algorithms during training as they are pre-trained, so it gives the best results at lower epochs only. When the model goes through each of the input images during each epoch it tries to track the common underlying patterns for the image with metallic surface cracks and tries to generalize the pattern by shifting the neuron weights. After training the model is tested on test set data which gives a validation accuracy of the model prediction. After 5 epochs we were able to attain 99% training accuracy and 94% testing accuracy. The accuracy and loss after each epoch are also plotted in the figures below.

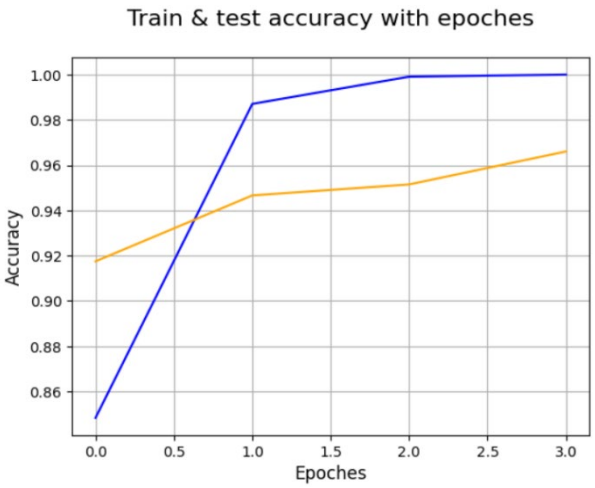

Fig. 9 Accuracy of model per Epoch VGG16

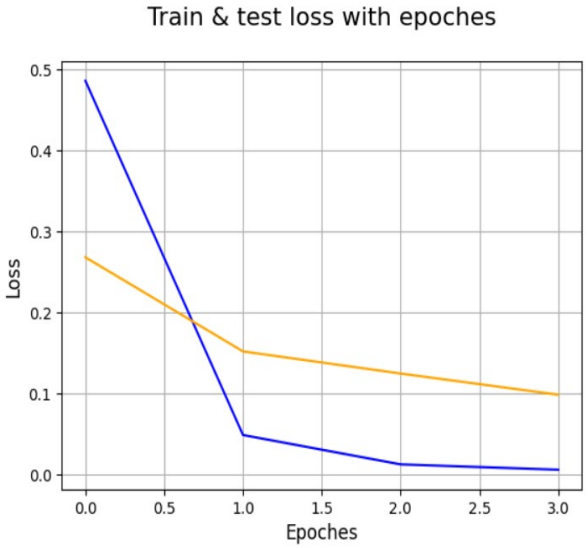

Fig. 10 Loss of model per Epoch VGG 16

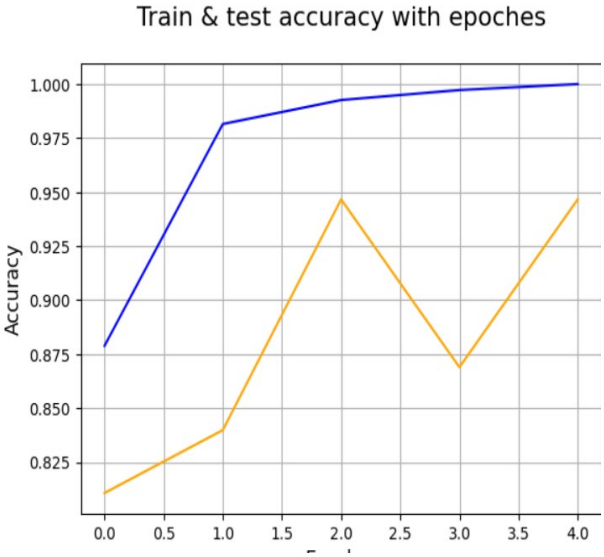

Fig. 11 Accuracy of model per Epoch Res152

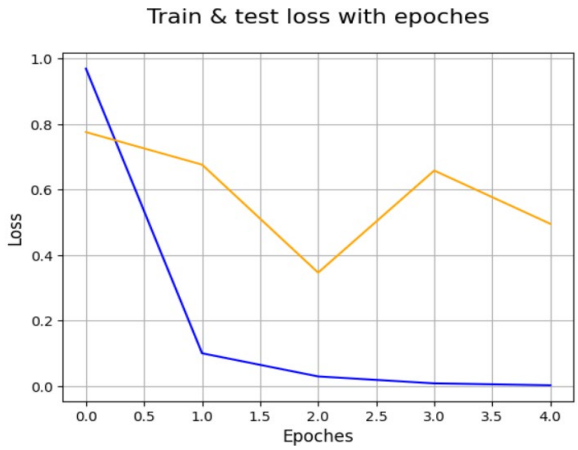

Fig. 12 Loss of model per Epoch Res152

TABLE I EVALUATION

Model	Accuracy	Precision	Recall	F1	Type 1 Error	Type 2 Error
VGG16	94.17%	0.9553	0.9386	0.9468	0.054	0.061
Res152	94.66%	0.9813	0.9210	0.9501	0.0217	0.079
MobileNet	78.64%	0.7822	0.8508	0.815	0.2934	0.149
AlexNet	61.23%	0.6287	0.7280	0.6747	0.5326	0.272
LeNet	47.09%	0.5196	0.5789	0.5474	0.6630	0.421

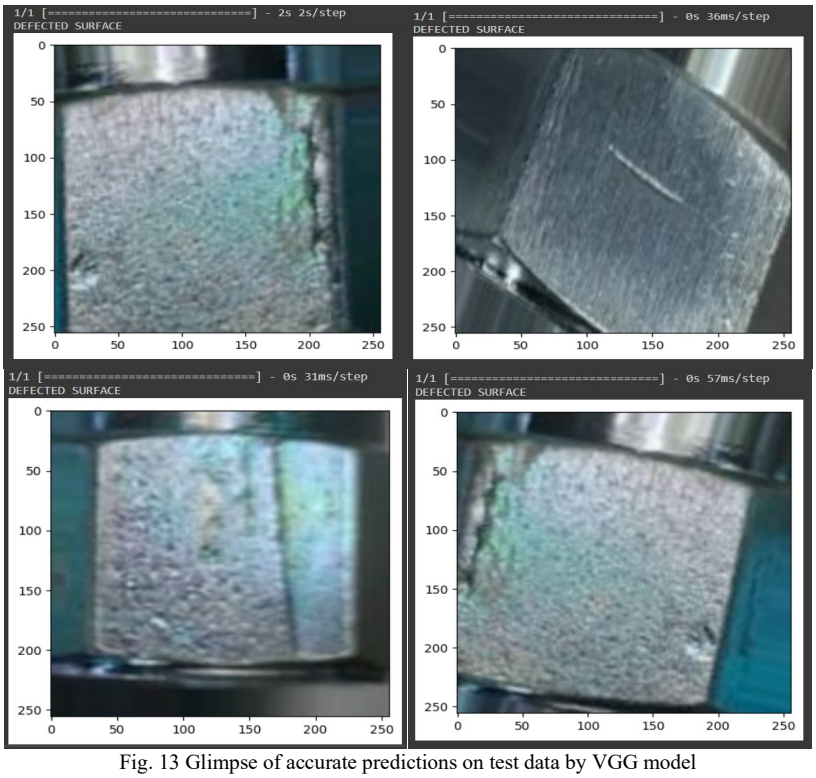

Fig. 13 Glimpse of accurate predictions on test data by VGG model

VGG16 and Resnet 152 outperformed the model expectation with around 94% accuracy in detecting faulty metal pieces. Mobilenet was able to gain accuracy of 78% but failed in cases of detecting sideline cracks in an input images. Alexnet and LeNet's, typical model architecture failed to generalize the underlying pattern for detecting the cracks during training of the model. Different model performances are evaluated using different parameters as presented in Table I. This completes model building, testing, and evaluation of the model architecture which can be deployed over and made live to detect and inspect the raw material at quality inspection. The results obtained in real time can be used to retrain the model to improve the model's ability to detect faulty pieces of different kinds.

VI. CONCLUSION AND FUTURE WORK

It was successfully suggested to use CNN-based deep learning to detect cracks in hex nuts. The CNN outperformed conventional edge detection methods with consistent performance across a range of settings and lighting conditions. It demonstrated potential in identifying tiny fissures under difficult illumination conditions. For this project, we used and compared various CNN architectures. The accuracy achieved with VGG16 is approximately 94.17%, with ResNet yielding 94.66%, MobileNet yielding almost 79%, AlexNet yielding approximately 61%, and LeNet yielding approximately 47%. The resilience of the technique was further shown by its independence from picture quality, camera specifications, and working distance. Comparative studies demonstrated its advantages over traditional techniques, especially under difficult conditions. The potential of CNN for a variety of damage detection applications is highlighted by its capacity to learn from large amounts of training data. Future work will integrate several data modalities, such as ultrasound or infrared imaging, to improve the comprehensiveness of fault detection. Fusion methods can take advantage of complementary information to increase robustness. Large datasets and pre-trained models combined with transfer learning could speed up training and improve generalization. For real-time detection in real-world applications, model architectures must be optimized for edge device deployment. Building trust is achieved by improving interpretability using gradient-based approaches or attention mechanisms. To perform predictive maintenance, it is necessary to comprehend the dynamic progression of problems. More extensive and varied datasets, cooperative benchmarking endeavours, and human-in-the-loop systems facilitate the enhancement and implementation of hexagonal defect detection models in practical engineering settings.

REFERENCES

[1] Ajmi, C., Zapata, J., Elferchichi, S., Zaafouri, A., & Laabidi, K. (2020). Deep Learning Technology for Weld Defects Classification Based on Transfer Learning and Activation Features. *Advances in Materials Science and Engineering, 2020,* 1–16. https://doi.org/10.1155/2020/1574350.

[2] Ali, R., Chuah, J. H., Talip, M. S. A., Mokhtar, N., & Shoaib, M. A. (2022). Structural crack detection using deep convolutional neural networks. *Automation in Construction, 133,* 103989. https://doi.org/10.1016/j.autcon.2021.103989.

[3] Bengio, Y., Courville, A., & Vincent, P. (2013). Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 35*(8), 1798-1828.

[4] Cha, Y., Choi, W., & Büyüköztürk, O. (2017). Deep Learning-Based Crack Damage Detection Using Convolutional Neural Networks. *Computer-Aided Civil and Infrastructure Engineering, 32*(5), 361-378. https://doi.org/10.1111/mice.12263.

[5] Cumbajin, E., Rodrigues, N., Costa, P., Miragaia, R., Frazão, L., Costa, N., Fernández-Caballero, A., Carneiro, J., Buruberri, L. H., & Pereira, A. (2023). A Systematic Review on Deep Learning with CNN Applied to Surface Defect Detection. *Journal of Imaging, 9*(10), 193. https://doi.org/10.3390/jimaging9100193.

[6] Hamishe Bahar, Y., Guan, H., So, S., & Jo, J. (2022). A Comprehensive Review of Deep Learning-Based Crack Detection Approaches. *Applied Sciences, 12*(3), 1374. https://doi.org/10.3390/app12031374.

[7] Ibrahim, A. A. M., & Tapamo, J. R. (2024). Transfer learning-based approach using new convolutional neural network classifier for steel surface defects classification. *Scientific African, 23,* e02066. https://doi.org/10.1016/j.sciaf.2024.e02066.

[8] Islam, M. M., Hossain, M. B., Akhtar, M. N., Moni, M. A., & Hasan, K. F. (2022). CNN Based on Transfer Learning Models Using Data Augmentation and Transformation for Detection of Concrete Crack. *Algorithms, 15*(8), 287. https://doi.org/10.3390/a15080287.

[9] Jha, S. B., & Babiceanu, R. F. (2023). Deep CNN-based visual defect detection: Survey of current literature. *Computers in Industry, 148,* 103911. https://doi.org/10.1016/j.compind.2023.103911.

[10] Jiang, Q., Tan, D., Li, Y., Ji, S., Cai, C., & Zheng, Q. (2019). Object Detection and Classification of Metal Polishing Shaft Surface Defects Based on Convolutional Neural Network Deep Learning. *Applied Sciences, 10*(1), 87. https://doi.org/10.3390/app10010087.

[11] Johnston, A. (2019). *Enhancing Structural Integrity: A Comprehensive Guide to Structural Health Monitoring.* Wiley.

[12] Konovalenko, I., Maruschak, P., Brevus, V., & Prentkovskis, O. (2021). Recognition of Scratches and Abrasions on Metal Surfaces Using a Classifier Based on a Convolutional Neural Network. *Metals, 11*(4), 549. https://doi.org/10.3390/met11040549.

[13] LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature, 521*(7553), 436-444.

[14] Li, S., & Zhao, X. (2017). A method of Crack Detection Based on a Convolutional Neural Network. Retrieved from https://www.researchgate.net/profile/Shengyuan-Li/publication/322840468_A_Method_of_Crack_Detection_Based_on_Convolutional_Neural_Networks/links/5c6a216b299bf1e3a5af0ad0/A-Method-of-Crack-Detection-Based-on-Convolutional-Neural-Networks.pdf.

[15] Muhammad, U., Wang, W., & Pervez, S. (2018). Pre-trained VGGNet Architecture for Remote-Sensing Image Scene Classification. *2018 24th International Conference on Pattern Recognition (ICPR).*

[16] Munawar, H. S., Hammad, A. W. A., Haddad, A., Soares, C. A. P., & Waller, S. T. (2021). Image-Based Crack Detection Methods: A Review. *Infrastructures, 6*(8), 115. https://doi.org/10.3390/infrastructures6080115.

[17] Piwal, H., Dhokale, M., Biswas, R., Raut, S., & Malge, A. (2023). Surface defect detection using deep learning. *International Conference on Smart Materials And Structures, ICSMS-2022.* https://doi.org/10.1063/5.0129207.

[18] Qayyum, W., Ehtisham, R., Bahrami, A., Camp, C., Mir, J., & Ahmad, A. (2023). Assessment of Convolutional Neural Network Pre-Trained Models for Detection and Orientation of Cracks. *Materials, 16*(2), 826. https://doi.org/10.3390/ma16020826.

[19] Qi, S., Yang, J., & Zhong, Z. (2020). A Review on Industrial Surface Defect Detection Based on Deep Learning Technology. *2020 the 3rd International Conference on Machine Learning and Machine Intelligence.* https://doi.org/10.1145/3426826.3426832.

[20] Sauter, D., Schmitz, A., Dikici, F., Baumgartl, H., & Buettner, R. (2021). Defect Detection of Metal Nuts Applying Convolutional Neural Networks. In *2021 IEEE 45th Annual Computers, Software, and Applications Conference (COMPSAC)* (pp. 248-257). Madrid, Spain. https://doi.org/10.1109/COMPSAC51774.2021.00043.

[21] Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv, 1409*(1556v6).

[22] Smith, J., Johnson, R., & Patel, K. (2020). Advances in Deep Learning for Image-Based Defect Detection: A Review. *IEEE Transactions on Industrial Informatics, 16*(1), 322-334.

[23] Soukup, D., & Huber-Mörk, R. (2014). Convolutional Neural Networks for Steel Surface Defect Detection from Imagemetric Stereo Images. *Lecture Notes in Computer Science.* https://doi.org/10.1007/978-3-319-14249-4_64.

[24] Venkatesh, R., Vignesh Saravanan, K., Aswin, V. R., Balaji, S., Amudhan, K., & Rajakarunakaran, S. (2022). Detection of Cracks in Surfaces and Materials Using Convolutional Neural Networks. *Mobile Radio Communications and 5G Networks,* 223-241. https://doi.org/10.1007/978-981-16-7018-3_18.

[25] Wu, F. (2022). Effect of transfer learning on the performance of VGGNet-16 and ResNet-50.

[26] Yi, L., Li, G., & Jiang, M. (2016). An End-to-End Steel Strip Surface Defects Recognition System Based on Convolutional Neural Networks. *Steel Research International, 88*(2). https://doi.org/10.1002/srin.201600068.