Deep Learning for Decay and Corrosion Detection in Industrial Environments

Dee Shi Ng1, 2, a), Jashila Nair Mogan1, 2, b), Sumendra Yogarayan1, 2, c)

1 Centre for Intelligent Cloud Computing, COE for Advanced Cloud, Multimedia University, Jalan Ayer Keroh Lama, 75450 Bukit Beruang, Melaka, Malaysia

2 Faculty of Information Science & Technology, Multimedia University, Jalan Ayer Keroh Lama, 75450 Bukit Beruang, Melaka, Malaysia

*b) Corresponding author: jashilanair@mmu.edu.my*

*a) 1211307066@student.mmu.edu.my*

*c) sumendra@mmu.edu.my*

**Abstract.** The maintenance and safety problem of detection of decay and corrosion is a serious matter. Extreme environments characterized by high temperatures, humidity, and chemicals as well as mechanical stress are common in the industrial setting like the oil rigs, power plants, factories and the transport infrastructure. All these contribute to the corrosion of materials that may lead to structural failures, business shutdowns, and safety risks. Factories have traditionally been using the manual internal inspection, as well as non-destructing testing methods to identify decay and corrosion. Nevertheless, such methods, are not always effective, not inexpensive and prone to a human error, especially in the context of large-scale or complicated industrial industries. This project is aimed at the implementation of the deep learning model to detect decay and corrosion in industry. As a baseline model, efficient architecture and good performance with classification problems, a pre-trained convolutional neural network, Xception, was selected. To increase its ability, more layers were added on the architecture to constitute the proposed model specific to the detection and classification of different types of surface degradation. To improve the performance of the model, fine tuning was done using a dataset of high-quality images which were clear and well defined to ensure more accuracy in identifying the various forms of decay and corrosion. The results of the experiments proved that the offered model had a high level of accuracy and that it was better than the classic approaches. This study indicates the promise of the deep learning to provide a realistic and feasible predictive maintenance method and early failure detection in the industrial environments.

# Introduction

Decay processes and corrosion are so dangerous processes of degradation as the causes of serious headaches in industrial settings. These processes lead to a vast variety of materials, primarily metals and concrete that causes shortened lifetime of equipment, structural weakness, extra cost of maintenance, and possible disastrous consequences in case of neglect. The electrochemical reaction of materials and the surrounding environment known as corrosion is so eminent in industries where there are chances of contact with moisture and chemical substances or air filled with salt like marine, oil and gas, and manufacturing industries among others. In the same manner, non-metallic materials, such as wood, polymers and insulation are liable to be decayed by environmental exposure, age, or microbial activity, and this may deteriorate further.

To achieve minimum downtime, cost-effective operations and maximum prolongation of the service life of industrial structures, it is first necessary to detect all forms of decay and corrosion in time and assess it properly. Traditional types of inspection are, however, labor intensive, time wasting and impractical in regions that are hazardous or difficult to reach into like underground or dungeons and also areas with heavy smokes. Traditional inspection is usually carried out manually through visual examination or contact methods. In addition, these methods are likely to give false results due to human errors and inconsistencies, thus cannot be used to monitor large numbers of people.

In a bid to counter such shortcomings, the deep learning concept designed and spearheaded by the convolutional neural networks (CNNs) has come forth with non-invasive, automated, and regularized inspection capabilities. These models do not only increase the accuracy of detection, but also bring the level of safety and efficiency as manual intervention is less required. Although there is considerable potential, most of the current research is either based on specific/superficial CNNs, or on approaches that remain unattested by exploration of pretrained models, which are designed to deal with the more complex and multifarious visual patterns that occur in real world industrial applications.

The current study aims at filling this gap as it proposes a model of deep learning based on an improved Xception architecture to enhance the measurement of decline and corrosion in non-managed industrial setups. The methodology trains the model with a wide range of data 1,819, web-scraped images of corrosion and no-corrosion with 990 and 829 images, respectively, in order to attain the robust and scalable performance that can be deployed in practice.

# Related works

Deep learning has transformed the solutions to issues encountered in industrial settings particularly where decay and corrosions are detected. It has been in the news lately because of its talents in automating the complicated inspection processes, with great precision. The models have the ability to detect and categorize decay and corrosion in industrial environments and make the process much simpler and without errors, which would have previously taken time. With the further development of the technology, new methods of dealing with the given issues, such as CNNs and Transformers are being developed [1], which only accelerates the process of detecting and combating them in an even more efficient and predictable manner.

Seddik et al. [2] have developed the model of detecting damage on pressure vessels based on the YOLOv8. The YOLOv8 model has a backbone, neck, and head structure where multi-scale features are extracted with the backbone, these features are refined and aggregated via the neck structure, and final output bounding boxes and class probability with the head structure used to localize and classify objects. Das et al. [3] used DenseNet121, EfficientNetB7, and ResNet-34 as encoder-decoder networks to train to build corrosion organs. DenseNet121 effectively identified reduced features in a dense manner giving it a high accuracy that did not require so much computation. EfficientNetB7 leveraged compound scaling for better performance, though its deeper versions overfitted the data due to excessive complexity. ResNet-34 utilized residual connections to stabilize learning and reduce errors. Li et al. [4] combined GAN with an enhanced YOLOv5-SE to detect corrosion in steel bridge bolts. The design of YOLOv5-SE architecture is to extract key features, reduce complexity and strengthen the important channels by using Squeeze-and-Excite (SE) layers.

Oluseyi et al. [5] introduced a multi-phase CNN framework; it designed to detect and identify the corrosion on metallic surfaces. This framework is the combination of deep learning techniques at different stages. Binary classification, multi-class categorization, and patch-based segmentation are included in it. Kapil et al. [6] introduced a deep learning-based system that used to archive the automate corrosion detection on steel bridge bolts. This system is based on a YOLOv5-SE model, which is a model customized version of YOLOv5 model. To strengthen the training process, the authors also incorporated a DCGAN that generates synthetic corrosion images, helping to expand the dataset.

Lu et al. [7] developed an FCM-LSTM model to detect anomalies in chemical tanks, combining clustering and LSTM for sequence analysis. The method integrates Long Short-Term Memory (LSTM) neural networks with Fuzzy C-Means clustering which using both supervised and unsupervised learning strengths to improve the detection accuracy. Forkan et al. [8] introduced CorrDetector, an artificial intelligence system that uses drone imagery and unifies deep learning techniques to improve corrosion detection. There are two convolutional networks include in this system. One of them is used to inspect industrial structures, another convolutional network is used to identify areas with significant corrosion.

Zhao et al. [9] compared EfficientNetV2 variants for binary classification of corrosion images, including B0, B1 and S. Among them, B0 provided the best balance between accuracy and resource efficiency. Recently, Huang et al. [10] introduced a machine vision-based framework to improve how corrosion is assessed in steel structures. This approach aims to address the inefficiencies and subjectivity that often exist in the manual inspection process. To simulate the developments of corrosion in real environment, the authors explored Q235 steel samples to salt spray tests.

While many existing studies have adopted CNN-based models for corrosion detection, most rely on custom or shallow architectures, limiting their ability to generalize across diverse and complex real-world conditions. Only a few works have explored the use of pretrained models for this task, suggesting an underexplored potential in leveraging deeper, transfer learning-based architectures. Moreover, current studies often depend on small or curated datasets, which may not represent the variability seen in industrial settings. This study addresses these gaps by proposing an enhanced Xception-based model, utilizing a more diverse and web-scraped dataset of 1,819 images, to improve detection robustness and performance in real-world corrosion scenarios.

# methodology

This study follows a structured methodology comprising four main components: dataset preparation, data preprocessing, model development, and experimental analysis. Images were sourced from publicly available platforms and labeled as *Corrosion* or *No Corrosion*, covering diverse industrial contexts such as ships, vehicles, and pipelines to ensure generalizability. The deep learning model was based on Xception architecture, with modifications including a custom classification head and optimized layers [11].

## Data Preprocessing

In this project, RGB images were used because they provide clear and detailed visual information that helps in identifying corrosion and other surface problems. Unlike grayscale images, which only show shades of grey, RGB images keep the full range of colours. This is important because things like rust, scratches, or paint damage often show up as subtle changes in colour or texture that might not be noticeable in black and white. In real industrial settings, where damage can look very different depending on materials, lighting, or how far along the corrosion is, keeping the full colour information makes the whole detection process more accurate and reliable. This makes RGB images not just a suitable option but a necessary one for achieving reliable results.

## Proposed Model

Xception, which stands for "Extreme Inception," is adopted as the main architecture to address the corrosion detection image classification task. The Xception model is designed as an evolution of the original Inception model, a key modification improved of Xception that enhances both efficiency and accuracy introduced. The Xception architecture consists of three main flows, namely the Entry Flow, the Middle Flow, and the Exit Flow. All the flows dwell upon feature extraction stage. The two are collaborating with each other to control how the input raw image is transformed into a detailed and meaningful one. These representations are then used by the model to make predictions on whether the imagery had signs of corrosion. Also, the network consists of a convolution layer and a separable layer covered by batch normalization layer. This may facilitate in stabilizing the learning process and accelerating the training.

In the case of the Entry Flow, that is the first stage of the Xception architecture and is concerned with the extraction of the basic visual features of the input image. There are two fundamental convolutional layers that comprise the process. The model may identify the lower visual things, such as edges and corners early on and flat, simple textures. Once the data has been extracted as the initial feature, it passes through a set of blocks. The blocks have a depth wise separable convolution and topping it with a max pooling. Such blocks include shortcut connections that were introduced through 1x1 convolutions with the kernel size of 2 which aid in matching features of different dimensions across layers. Also, residual connections are added throughout the model to maintain a stable training. They aid uniform gradient flow in the backpropagation, and they avoid the issue of vanishing gradients at deeper levels.

As data progresses deeper into the network, Middle flow keeps extracting features by discovering higher-level patterns that have never existed in the previous phases. This component of the model is central and has the role of learning more profound and advanced characteristics. It consists of some block that is repeated eight times. In every block, the model uses three depth wise separable convolutional block, then a batch normalization and a non-linear activation block. An interesting design feature within Xception is the fact that no activation function is used between the depth wise convolution and the point wise convolution. Such transformations as ReLU or ELU are not applied deliberately at this level. It can assist the model to train quicker and create more productive overall performance. The feature maps retain the spatial dimensionality which is done during the Middle Flow. This enables the model to explore more meaningful patterns in the input data while keeping the computational cost. The repetitive block structure further enables the extraction of rich hierarchical features that are crucial for distinguishing the subtle differences between corrosion and no corrosion.

After that, the Exit Flow continues the feature refinement process based on what the model has learned in earlier stages. This stage of architecture combines additional separable convolutions with many filters. The added layers help the network focus on finer details and discover deeper information in the input image. Before the data goes into deeper layers, there is a 1×1 convolution is applied for the purpose of aligning the feature dimensions. In this flow, there is another max pooling operation that is used to further reduce the spatial resolution. After this, there are two separable convolution layers with 1536 and 2048 filters applied one after the other. These powerful layers help the model bring together everything it has learned so far into a compact form that still captures a lot of important details. Afterwards, there is a global average pooling layer that is used to convert the feature map into a single vector representing each class. The last stage of the architecture consists of a classification layer. There is a SoftMax activation function applied at this stage to produce the probability distribution across the output classes. This enables the model to classify each input image as either "Corrosion" or "No Corrosion" category.

The proposed model is developed on top of the pre-trained Xception architecture. Figure 1 shows the architecture of the proposed model. First, there is a Global Average Pooling layer added to simplify the feature maps by reducing their dimensions after the model extracts important features from the input images, this making the data easier to handle in the following layers.

Next, a dense layer with 256 units and ReLU activation function is added. The ReLU activation function is used to introduce nonlinearity that can help the model learn complex patterns. To reduce the probability of overfitting, a dropout layer with a rate of 0.4 was added immediately after this. This helps the model stay more general and avoid memorizing the training data too closely. Then, another dense layer with 128 units and ReLU activation function is added, followed by a same dropout layer to further fine-tune the learned features. Finally, the SoftMax output layer is added to generate probabilities for each class to determine whether the image shows corrosion. The model with these added layers enhances the model’s ability to learn and recognize specific patterns in the dataset, this makes it more accurate and reliable when applied to actual corrosion detection tasks.

A screenshot of a diagram

AI-generated content may be incorrect.

**FIGURE 1.** Architecture of the Proposed Model

# Experiments and discussion

Initially, the proposed model was trained over a total of 100 epochs. During the entire training process, the accuracy and loss values for the training and validation sets were monitored closely to evaluate the learning effect of the model.

During the initial stages of training, the model achieved a moderate accuracy of approximately 66.64%. As training progressed, performance improved steadily. At the end of the training phase, the model achieved a peak training accuracy of 95.76%. For the validation set, the accuracy gradually increases from 89.56% and finally reaches a peak of 94.51%. Throughout the training process, both the training and validation loss values ​​continue to decrease, indicating that the model is learning more effectively with each successive epoch.

To prevent overfitting, an early stopping function has been implemented. This technique monitors the validation loss and will stop immediately when there are no improvements within the epoch set. This can prevent the model from overtrain of start memorizing the data.

When tested on a separate dataset, the model achieved a good accuracy of 96.15%. Although the results are very promising, slight fluctuations in validation accuracy were observed during training. These changes in performance can be early warning signs that your model is beginning to be overfitting. Early stopping technique was employed to mitigate the model from overfitting.

## Dataset

The dataset consists of 1819 images in total, dividing 990 images as corrosion and 829 images as no corrosion. Web scraping technique was utilized to collect the images, which mainly collected from Google Images. Curated manuals were then done to ensure the images are pertinent to the corrosion detection task.

The corrosion category includes various examples of real-world damage that is visible including images that showing the rust on ship hulls, corroded ship propellers, rusted car bodies and corrosion in oil and gas pipelines. On the other hand, the no corrosion class features contain with the image that clean or undamaged surfaces from similar environments and objects. This can help the model learn to distinguish between the features of corrosion and no corrosion.

Figure 2 illustrates the sample images from both the corrosion and no corrosion classes. These figures provide a visual reference to understanding the types of surfaces, damage, and environmental conditions the model was trained to analyze.

|  |  |
| --- | --- |
| Close-up of a tree trunk  Description automatically generated | A close-up of a silver metal  Description automatically generated |
| (a) Corrosion | (b) No Corrosion |

**FIGURE 2.** Example of a Dataset Image in the (a) Corrosion Class, (b) No Corrosion Class

## Hyperparameter Tuning

There are seven key hyperparameters that need to be fine-tuned in this work, such as batch size (B), learning rate (R), input size (I), optimizer (θ), dropout rate (D), and activation function (A). These elements were fine-tuned based on their individual and combined impacts on model accuracy with the goal of identifying the most effective configuration for this specific classification task.

Table 1 shows a comparison of different batch sizes. As a result, the best performance among them is 96.70%, which is produced by the 32-batch size. For the smaller batch size, it typically introduces more frequent updates to the model’s weight that allows for finer adjustments during training. This is particularly useful when the dataset contains subtle variations, such as when distinguishing between corrosion and no corrosion surfaces. When it was adjusted to 64, accuracy dropped slightly to 94.51% and further declined to 92.86% with a batch size of 128. These results suggest that larger batch sizes may overgeneralize the learning process and may ignore fine-grained features required for accurate classification.

|  |  |  |
| --- | --- | --- |
| **TABLE 1**. Accuracy at Different Batch Sizes B [I =224 x 224, θ = Adam, R = 0.001, D = 0.4, A = ReLU] | | |
| **Batch Size** | **Accuracy** |
| 32 | 96.70% |
| 64 | 94.51% |
| 128 | 92.86% |

Table 2 shows a comparison of different optimizers. When evaluating different optimizers, Adam optimizer consistently outperformed the others. Adam achieved an accuracy of 96.70%, which was the highest recorded in this study. Adam's success is due to its adaptive learning rate mechanism, which dynamically adjusts the step size during training to help the model converge more efficiently. The Stochastic Gradient Descent (SGD) optimizer came in second with 95.05%, while Nadam, a variant of Adam with Nesterov momentum, resulted in 93.96% accuracy. While all three optimizers performed reasonably well, Adam's combination of speed and adaptability made it the most effective for this task.

|  |  |  |
| --- | --- | --- |
| **TABLE 2.** Accuracy at Different Optimizers θ [*I* =224 x 224, B = 32 x 32, R = 0.001, D = 0.4, A = ReLU] | | |
| **Optimizer** | **Accuracy** |
| Adam | 96.70% |
| SGD | 95.05% |
| Nadam | 93.96% |

Table 3 shows a comparison of different learning rates. When the learning rate is 0.001, the performance is the best, with an accuracy of 96.70%. This shows that a slower learning process allowed the model to gradually fine-tune its weights and avoid overshooting the optimal solution. The learning rate of 0.01and 0.0001 producing accuracies of 91.21% and 95.6%. The model trained with a learning rate of 0.1 performed poorly, reaching only 68.13%. This was likely due to the model skipping over optimal points during training, which is a common issue with excessively high learning rates.

|  |  |
| --- | --- |
| **TABLE 3.** Accuracy at Different Learning Rate R [I =224 x 224, B = 32 x 32, θ = Adam, D = 0.4, A = ReLU] | |
| **Learning Rate** | **Accuracy** | |
| 0.0001 | 95.60% | |
| 0.001 | 96.70% | |
| 0.01  0.1 | 91.21%  68.13% | |

Dropout is a regularization technique used to prevent overfitting by randomly deactivating a portion of neurons during training. Table 4 shows a comparison of different dropout rates. In this experiment, a dropout rate of 0.4 produced the highest accuracy, scoring 96.70%. This shows that regularizing the model at a relatively high rate helped improve its robustness and avoid memorizing training data. In contrast, lower dropout values such as 0.2 and 0.3 led to lower performance, yielding accuracies of 93.96% and 95.05%, respectively. These results suggest that while some regularization is helpful, stronger dropout in this case is more effective in reducing overfitting without hampering the model's ability to learn.

|  |  |
| --- | --- |
| **TABLE 4.** Accuracy at Different Dropout Rate D [*I* =224 x 224, B = 32 x 32, θ = Adam, R = 0.001, A = ReLU] | |
| **Dropout Rate** | **Accuracy** | |
| 0.2 | 93.96% | |
| 0.3  0.4  0.5 | 95.05%  96.70%  96.15% | |

For the activation function, ReLU and Leaky ReLU were tested. Although Leaky ReLU achieved relatively high accuracy of 95.05%, the standard ReLU activation function ultimately produced the best performance in this fine-tuning experiment, reaching an accuracy of 96.7%. These results suggest that while Leaky ReLU can be effective, its performance is sensitive to slope adjustments, and ReLU remains the more robust choice in this context. Table 6 shows the accuracy achieved by each tested activation function.

|  |  |
| --- | --- |
| **TABLE 6.** Accuracy at Different Activation Function A [I =224 x 224, B = 32 x 32, θ = Adam, R = 0.001, D = 0.4] | |
| **Activation** | **Accuracy** |
| ReLU  Leaky ReLU | 96.70%  95.05% |

Table 7 shows a comparison of the different input sizes. The model achieved the highest accuracy of 96.70% at 224 × 224 pixels, suggesting this resolution offers an optimal balance between detail and computational load. An input size of 128 × 128 resulted in a reduced accuracy of 91.21%, likely due to insufficient texture information. Conversely, the accuracy slightly dropped using input size of 256 × 256, possibly due to the introduction of an additional noise. These findings demonstrate the importance of selecting an appropriate input size to optimize the model performance.

|  |  |  |
| --- | --- | --- |
| **TABLE 7.** Accuracy at Different Input Sizes I [B = 32 x 32, θ = Adam, R = 0.001, A = ReLU, D = 0.4] | | |
| **Input Size** | **Accuracy** |
| 128 x 128  224 x 224  256 x 256 | 91.21%  96.70%  95.60% |

The ideal hyperparameter values are including batch size of 32, dropout rate of 0.4, input size of 244 x 244, Adam optimizer, learning rate of 0.001, and ReLU activation. This setup obtained the highest accuracy, which makes the model more reliable for practical deployment in industrial corrosion detection applications.

## Comparative Analysis with Existing Methods

Table 8 displays a performance comparison with existing methods. The existing works yielded promisingly with an accuracy above 90%. Nevertheless, the proposed model has shown a superior performance with an accuracy of 96.70%. The improvement is attributed to enhancements made to the Xception architecture, the integration of two additional fully connected layers and dropout regularization. These architectural changes significantly boost the model's ability to capture complex and abstract features, which is especially valuable when working with noisy and diverse images.

Maximisation of parameters is done by adding fully connected layers. This enhances complex learning representations. Simultaneously, the dropout regularization allows to decrease the problem of overfitting. It counters the tendency of neurons being over reliant upon each other when training to enhance usefulness on new information. Notwithstanding that these improvements increase the computation cost, the increases in accuracy and robustness justify the trade-off.

Although the new model introduces additional complexity over the previous methods the classification improvement and generalization are high, and the model would be a good contender in using it on practical industrial corrosion detection projects.

|  |  |
| --- | --- |
| **TABLE 8.** Comparative Analysis of Corrosion Detection Methods | |
| **Methods** | **Accuracy** |
| FCM-LSTM [7]  EfficientNetV2-B0 [9]  InceptionV3 [10]  Proposed Model | 95.87%  91.76%  93.41%  96.70% |

# Conclusion

Finally, as compared to the existing approaches to the problem of industrial corrosion detection, the proposed model reveals a higher accuracy rate of 96.70 percent. It is a tremendous increase, which demonstrates the effectiveness of the redesigned model and selected training strategy, indicating its potential use in real industrial conditions. The model found that the obtained results confirm its capacity to offer valid and precise results that are needed in schedules of maintenance and guaranteeing security in the provision in infrastructure and manufacturing environment. Future research will consist in improving the dataset by adding a greater amount of varied and representative sampling so as to further the generalization and reliability of the model.

# REFERENCES

1. S.C. Joseph, G. S., K. Suriyan, and S.P. K., “Performance Benchmarking: Pre-trained Models and Custom Convolutional Neural Networks in Deep Learning,” Journal of Informatics and Web Engineering **4**(2), 225–235 (2025).
2. E, Seddik, M. Hewieg, and R. Afify. “Artificial Intelligence (AI) Powered Detection and Identification of Pressure Vessel External Damage–YOLO-based Application”, (2024).
3. A. Das, S. Dorafshan, and N. Kaabouch, “Autonomous Image-Based Corrosion Detection in Steel Structures Using Deep Learning,” Sensors **24**(11), 3630 (2024).
4. Z. Li, P. Shao, M. Zhao, K. Yan, G. Liu, L. Wan, X. Xu, and K. Li, “Optimized deep learning for steel bridge bolt corrosion detection and classification,” Journal of Constructional Steel Research **215**, 108570 (2024).
5. O.A. Oyedeji, S. Khan, and J.A. Erkoyuncu, “Application of CNN for multiple phase corrosion identification and region detection,” Applied Soft Computing **164**, 112008 (2024).
6. K. Mundada, M. Kulkarni, P. Sagar, R. Asegaonkar, and N. Mulgir, “Corrosion Detection using Deep Learning and Custom Object Detection,” in *2022 2nd Asian Conference on Innovation in Technology (ASIANCON)*, (IEEE, Ravet, India, 2022), pp. 1–5.
7. X. Lu, Y. Lin, and L. Wang, “Research on Anomaly Detection Algorithm of Chemical Storage Tanks Based on FCM-LSTM,” in *2023 International Conference on Power, Electrical Engineering, Electronics and Control (PEEEC)*, (IEEE, Athens, Greece, 2023), pp. 720–726.
8. A.R.M. Forkan, Y.-B. Kang, P.P. Jayaraman, K. Liao, R. Kaul, G. Morgan, R. Ranjan, and S. Sinha, “CorrDetector: A framework for structural corrosion detection from drone images using ensemble deep learning,” Expert Systems with Applications **193**, 116461 (2022).
9. Z. Zhao, E.B.A. Bakar, N.B.A. Razak, and M.N. Akhtar, “Corrosion image classification method based on EfficientNetV2,” Heliyon **10**(17), e36754 (2024).
10. X. Huang, Z. Duan, S. Hao, J. Hou, W. Chen, and L. Cai, “A Deep Learning Framework for Corrosion Assessment of Steel Structures Using Inception v3 Model,” Buildings **15**(4), 512 (2025).
11. F. Chollet, “Xception: Deep Learning with Depthwise Separable Convolutions,” in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (IEEE, Honolulu, HI, 2017), pp. 1800–1807.