Image Processing for Industrial Structure Analysis

Sérgio Neves
Instituto Superior Técnico - Universidade de Lisboa

***Abstract*—As the world becomes increasingly more dependent on industrial infrastructures, structures like electrical power towers are very important for society to function properly, as the failure of these kinds of structure could affect thousands of people. This makes the inspection and maintenance of these structures a necessity.**

The most common way of analysing structures, like electrical towers, is by a human inspector visually assessing the state of the structure, which is time consuming, expensive and potentially dangerous. However, with the increased usage of robotic systems, such as drones, which are unmanned areal vehicles, and high resolution cameras, it is possible to acquire images from structures in a faster and safer way. With this information it is possible to further analyse the images obtained from these structures in order to detect which images contain a significant amount of rust and should be evaluated by an expert.

This thesis proposes the construction of two different methodologies for detecting the amount of rust in structures, the first is an adaptation of a technique based on a sliding window and utilizing a convolution neural network for detecting rust regions, and the second is a methodology based on semantic segmentation for detecting how many pixels in an image are detected as "rust" and how many are detected as "structure". These two approaches will then be compared in order to determine their advantages and limitations, when trying to achieve a good performance for the desired task.

I. Introduction

Rust development in industrial infrastructures, such as electrical towers for energy transportation, telecommunication towers and wind turbines, is responsible for deterioration and subsequent failure (such as power outages) of these types of structures. The corrosion of structures also poses a large economic cost, being estimated that the global cost for corrosion in 2013 was US$2.5 trillion, which is 3.4% of the global gross domestic product (GDP) [1].

Besides the large economic aspect of industrial structure maintenance, since the most common practice is to deploy a person or group of people to the field in order to assess the condition of the structure, this also raises concerns about the safety of those involved in the inspection task, as these people may need to climb the structures to evaluate their conditions and often make a second climb when it is determined that some kind of intervention is needed to fix the corroded or rusted area.

In recent years, with the development and usage of aerial vehicles, such as drones, it is possible to reduce the amount of personnel that is put in hazardous conditions for performing the inspection of infrastructures. Nonetheless, the amount of work that specialists must undertake in order to detect dangerous levels of rust in industrial structures could still be further reduced. Even though the usage of drones reduces the amount of time necessary to access the state of a structure, there are still a large amount of captured images that is. Therefore, these workers must still go through large amount of irrelevant images of the structure captured by the drone to identify structure areas that need maintenance interventions.

This project analyses several methods of detection of rust in images. The goal is to develop a system that is capable of making an automatic detection of rust in the imaged structure, as a way to reduce the amount of images that an expert needs to go through when assessing the state of industrial structures. The images are taken from different distances and on structures with varied background, which increases the difficulty of detecting regions which regions are structure and which regions are rust.

II. Related Work

The main effect of corrosion is that, over time, it can cause structural flaws, making the structures more fragile which in turn can lead to the malfunctioning or even the fall of electrical towers. Structural flaws due to corrosion are characterized differently depending if they affect the surface or the interior of the structure, Table I depicts some common methods used in nondestructive structure evaluation for the detection of rust.

Evaluation Methods	
Surface Rust	Interior Rust
Visual	Magnetic field
Liquid Penetrant	Radiography
Eddy current	Energy dispersive X-rays
Magnetic particle	Microwave
Ultrasonics	Ultrasonics
Acoustic emission	Acoustic emission

Table I: Table with examples of non-destructive rust detection and characterization methods (adapted from [4]).

This work will focus on exploring several methods based on visual inspection, which is one of the most unintrusive methods and is commonly employed as the first step in evaluating a rusted area. Even though visual inspection does not give the most accurate depiction of the real condition of the structure, a majority of the time a visual inspection is done as a preliminary analysis of the structure condition.

There are several aspects that visually indicate the presence of corrosion, for example, in the case of rust, it can easily be seen that rusting leaves a clear red coloration on steel and creates coarser texture when compared to non-corroded areas. Looking at these characteristics, it is possible to see

how methodologies based on color, texture and shape could be used for detecting rust.

There have been several studies for detection and classification of objects, people and animals in images [2, 3, 4, 5, 6, 7], however the number of these studies that focused on image-based corrosion detection is limited. Often image-based rust detection systems rely on some sort of color analysis, a combination of color and texture analysis, possibly resorting to deep learning approaches [4, 8, 9, 10]. Other systems use techniques based on shape detection, such as using a stereo-adapter on a regular camera to measure the 3D shape of the corroded area to detect subsurface corrosion [11], or a system focusing on the pitting corrosion of aluminum, depending on the optical contrast of the corroded regions when compared to their surroundings and using edge detection to compute each closed and corroded region [12], these methods have some problems that make them inapplicable for this project. In the first case, subsurface corrosion can be identified by changes to the surface shape [8], this method needs, as mentioned previously, a camera with a specific adaptor, which is not ideal, given that the goal is to use footage captured by drones. As for the second approach, pitting is not a dominant feature of most types of regular corrosion, being a more common occurrence in aluminium corrosion, in particular rust's dominate features tend to be discoloration and texture degradation [4, 9], as such this technique is very specific and not useful for rust detection.

A. Color Analysis

What first jumps to mind when thinking about the physical characteristics of rust and corrosion in is the discolouration. Having said this it is no wonder that many of the first works on corrosion detection were based on color analysis. These techniques showed some promising results, but had some underlying flaws.

Some research done using that are based on color analysis include Petricca's *et al.* [10] by detecting rust using a range of intervals for rust in the hue, saturation, value (HSV) color space was defined, and later the image was segmented, to create a black and white mask, where the white pixels would correspond to the positions in the image where rust was detected, however this method proved to be ineffective due to how easily it misclassified objects (such as apples) as rust.

Another work that also used the HSV color space was done by Furuta [13], in his approach a neural network (NN) was used in order to calculate the threshold of hue and value in this color space that indicated that the image contained rust.

Lee [14], used a statistical analysis of the red, green, blue (RGB) color space in order to classify images taken from a bridge as defective or non-defective. This was done by plotting the images in the RGB color space, on which was later performed statistical analysis to obtain relevant statistical data or variables. This data and variables were then used to create multivariate discriminate models that were latter used to classify images as defective or non-defective.

B. Color and Texture Analysis

The approaches that are based on color and texture analysis comprise the majority of image based corrosion detection techniques and these can be further divided into statistical and filter-based approaches [9].

Statistical Approaches: A great number of the statistical based approaches can be seen utilizing gray level co-occurrence matrices (GLCMs), which give information on how often different combinations of pixel gray levels occur, mainly to extract statistical texture features [15]. These works typically use these GLCMs to obtain texture information and statistical analysis on HSV or hue, saturation, intensity (HSI) color spaces, similar to what is done in color analysis.

Taking as an example Medeiro's *et al.* [16] paper, in it GLCMs are used to obtain several texture features or attributes, from which four were utilized: contrast, correlation, energy, and homogeneity. Then the HSI color space was used to obtain color features. From these it was defined that: the hue (H) for a corroded surface lies between yellow and red wavelengths; the saturation (S) of a corroded area tend to be higher than other areas; metallic surfaces are usually gray or white, tending towards a whiter wavelength, which in turn leads to high intensity (I). By using a combination of the subsets of color and texture features, it was verified a better performance when compared to the results obtained from each individual subset.

Choi [17] created a system that was also based on obtaining color features through the HSI color space and GLCMs for obtaining texture features. In this approach, principal component analysis and varimax techniques were applied to the HSI color space in order to optimize the set of attributes, while eliminating the insignificant ones and minimize even further the attributes. For obtaining texture features, a method of GLCMs based on the azimuth difference of points on the surface was used.

Some downsides of using GLCMs, as pointed out by Xie [18] include the challenge of reducing the number of gray levels to keep the co-occurrence matrix from being too big of a computational burden, finding a number of entries for the matrix that maintains statistical reliability and optimizing the displacement vector.

Filter Based Approaches: Filter based approaches, as the name implies, are based on applying filter banks to the image and computing the energy of the response to those filters [18]. In corrosion detection, the most relevant and widespread techniques that utilize filter banks rely on the wavelet transform[9]. This is due to the wavelet transform having the capability to provide understanding in both the spatial and frequency domain of an image [9, 18]

Ghanta [19] develop a system for rust defect detection on steel bridges based on using Haar wavelet and principal component analysis, in order to design a classifier based on the least mean

Jahanshahi [9] expanded the research done on corrosion detection using wavelets by applying depth perception and different color spaces in order to improve reliability and performance. This work also explored the effects of different

color spaces and it was found that the color space that performed the best for texture analysis was the chroma blue-difference/chroma red-difference (CbCr) color space. The classification of the regions as corroded or non-corroded is done through a NN, where the inputs are the features computed based on the wavelet filter bank. This method obtained better results than any of the previous techniques, being, in essence, an improvement on the previous wavelet-based approaches.

C. Deep Learning Approaches

Rust detection is a niche field when compared to other types of object detection, meaning that the amount of works that focus on deep learning based approaches for rust detection is quite small. Nonetheless, there have been some studies realized in this area the show the effectiveness of these kinds of approaches.

Petricca *et al.* [10] performed comparisons between color and deep learning based approaches. The deep learning method used is based on using a pretrained model of the AlexNet [20], this network showed how promising convolutional neural networks could be for image classification when it won the ImageNet [21] challenge in 2012. As it can be seen, in Table II, the deep learning approach in almost ten percent more effective than the color based one when comparing the total accuracy.

	Color	Deep Learning
False Positive	27/63	14/63
Partial accuracy for "non-rust"	57%	78%
False Negative	4/37	8/37
Partial Accuracy for "rust"	89%	78%
Total Accuracy	**69%**	**78%**

Table II: Color and deep learning method comparison (adapted from [10]).

A method that has shown to surpass the previous state of the art developed by Jahanshahi, is the one developed by Atha [4]. This method is based on the use of a pretrained convolutional neural network (CNN) that showed good performance in the ImageNet challenge, the VGG [22] networks, and fine-tuning it to better detect corrosion. This trained network was used to classify sections of the image obtained from a sliding window. Comparing the results obtained from this approach with the wavelet and neural network approach developed by Jahanshahi can be seen in Table III. In this Table the overall superiority of using a CNN can be seen, as it is able to beat the previous state of the art in recall (the amount of regions correctly classified as corrosion divided by the sum of regions correctly classified as corrosion and regions wrongly classified as non-corrosion), precision (the amount of regions correctly classified as corrosion divided by all regions classified as corrosion) and F1 score (the harmonic mean of recall and precision).

Taking into account the results of this research, it can be concluded that methodologies based on deep learning approaches have proved highly effective and were even able to surpass previous state of the art approaches. Given this, in this work the problem of rust and structure detection will be tackled by using two different approaches. The first approach is based on utilizing a standard CNN to detect rust and structure in an image resorting to a sliding window and classifying each window. The second is based on using a SegNet architecture to segment the image into three possible colors, which represent, "rust" , "structure" and "background".

Algorithm	Recall (%)	Precision	F1 Score (%)
VGG16 (RGB)	98.31	98.64	98.47
Wavelet NN (RGB)	93.01	93.65	93.32
Wavelet NN (YCbCr)	85.41	89.85	87.53
Wavelet NN (CbCr)	75.85	84.87	80.03

Table III: Deep learning performance compared to the performance of wavelet and neural network for corrosion detection (adapted from [4]).

III. Deep Learning

A. Convolutional neural network

A convolutional neural network is usually composed by convolution layers, pooling layers (usually max pooling) and fully connected layers (usually these layers contain the rectified linear unit (ReLU) function as an activation function), where the last fully connected layer, also known as the output layer, commonly contains a different activation function, like the *sigmoid* or *softmax* functions. The basic layout of a CNN can be seen in Figure 1.

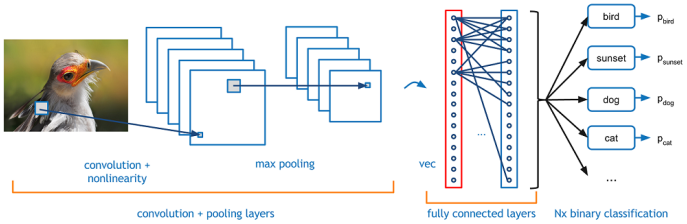


Figure 1: Structure of a standard CNN (adapted from [23]).

B. SegNet

SegNet is an architecture developed for semantic pixel labelling [7]. Meaning that it, given an image as an input, it outputs said image divided into different segments, where each of these segments represent a class. It is based on an encoder-decoder network, as seen in Figure 2, where the encoder is composed by convolutions and max pooling, while the decoder is composed of upsampling and convolutions, terminating by using a *softmax* classifier, that classifies each pixel. During the max pooling process, the max pooling indices or locations are stored and are then used during the upsampling process.

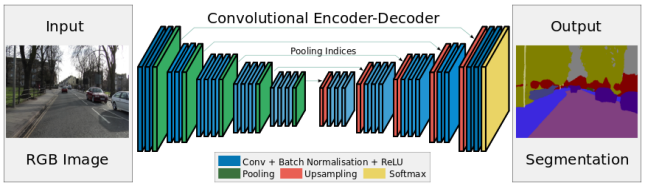


Figure 2: SegNet architecture (taken from [24])

IV. Architecture Development

The objective of this work is to construct a tool that is capable of reducing the amount of images that experts have to look at in order determine the corrosion state of a structure. For this, it is necessary to detect the presence of rust in an image. However, not all images that contain rust provide useful information for experts, since some of them only show a small area of rust. As such this work proposes a methodology that takes into account the relation between the areas of rust and structure found in an image.

A. Sliding Window and CNN Approach

This approach is inspired on Athas' developed system [4], changing the number of classes that the CNN will classify, making the network classify windows as "rust", "structure" or "background, instead of a binary classification of "rust" or "non-rust".

***System Architecture**:* Figure 3, shows a high level overview of the proposed system architecture based on utilizing a sliding window to classify segments of the image, which is comprised of four main components: pre-processing, feature extraction, classification and result analysis. The original input image goes through a pre-processing step in order to make it usable by the feature extractor, that is based on a CNN approach. This feature extraction obtains information on the image features, which in turn is used in order to classify the input of the CNN as "rust" or "structure". Afterwards the results obtained from the classification step are analysed to make a final classification.

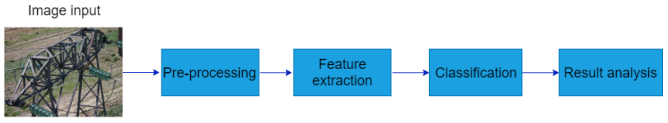


Figure 3: Architecture of the developed system based on a sliding window

***Pre-processing**:* The original data set is labeled by masks that label each pixel of the original mask as "rust", "structure" or "background". In order to transform this data set into something that can be used by a CNN, images with single labels were created by dividing the original and mask images into several blocks or windows - an example of this is shown in Figure 4, the label of each window will be one of the three possible classes: "rust", "structure" or "background".

The main issue to take into account when generating the window based ground truth data set from the available data, is how the rust and structure labeling is defined. Athas' [4] approach used images that where 100% rust in order to train the CNN used. However the percentage of rust in an image will affect what the CNN learns, as such testing different amounts of "rust" and "structure" percentage is important in order to choose the values that best represents these classes. In this project the vectors utilized will be $sp = [40, 60, 80, 90]$ and $rp = [25, 50, 75, 90]$.

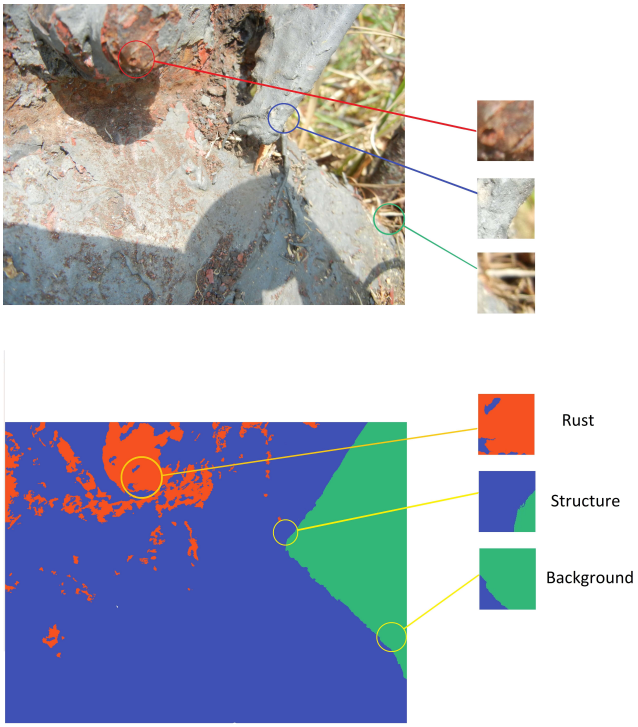


Figure 4: Example of a window from an image (original images provided by Axians).

After labeling the image windows into the three distinct classes, it is noticeable that the number of "background" and "structure" classes extremely out number the amount of images belonging to the "rust" class. This poses a problem, as the goal is to be able to classify correctly rust. But, if this class is so underrepresented, it will most likely cause the system to be more biased towards learning the "structure" and "background" classes, increasing the accuracy for those classes, and decreasing the accuracy for the "rust" class. Undersampling is done to reduce the amount of samples until all classes have the same amount of images, as seen in Figure 5.

In this project the accuracy of all classes is equally important. It is not desirable to misclassify any classes, because this will affect the calculation of the ratio between "rust" and "structure" classes for the entire image. However, since the amount of data is quite limited and somewhat repetitive, oversampling might lead to an increase in overfitting. This means that applying undersampling to the data is the adopted option.

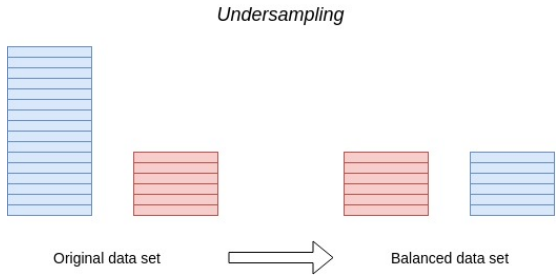


Figure 5: Example of undersampling.

Feature Extraction: For this approach, a convolution neural network as a means of extracting the features of an image, that, has discussed previously, as already shown good results in detecting rust in images [4]. Since CNN learn the important features of the classes they want to learn by themselves during training, it is possible to extent these methodologies from a binary classification of ”rust” and ”non rust” to a categorical classification of ”rust”, ”structure” and ”background”.

It was decided to implement the CNN resorting to the Keras [25] framework, which simplifies the implementation of this type of approach when compared to other frameworks, such as Tensorflow [26] and Pytorch [27]. For this project, two CNN architectures were tested, the VGG16 [22], that already showed promising results classifying rust [4] and the MobileNet [28], that is a more lightweight CNN capable of being implemented in mobile and other less powerful devices.

Result Analysis: The objective of this project is to filter the amount of images that experts have to look through. As such only images that have a relevant amount of ”rust” should be shown and images that only show ”background” or ”structure” but little or no ”rust” should not be shown. This, however, poses the following challenges:

- How much rust should be detected in an image for it to be considered relevant?
- How to obtain the amount of rust existing in a given image?

Regarding the first challenge, if rust is detected in a very small section of the image (only a *64x64* square is classified as rust in a *6000x4000* image, for example) it does not make sense to consider it relevant. This means that a threshold must be set in order to define whether the amount of rust present in a certain image is significant or not. of rust contained in the structure was defined as the cut off point for determining if an image was sufficiently corroded.

As for the second challenge, the ratio chosen for this evaluation is given by

$$\frac{n^{o}\ of\ rust\ windows}{(n^{o}\ of\ rust\ windows) + (n^{o}\ of\ structure\ windows)} \quad (1)$$

The ratio defined in 1, will define how much of the structure present in a given image was detected as being rust.

Utilizing the values stored in the *rust_count* and *structure_count*, a ratio of the amount of rust existing in the structure can be obtained. This can be done by applying the formula defined in 1. Having this ratio value it is possible to compare it to a threshold previously set for this ratio. For example, it could be defined that an image that shows a ratio of less than 2% rust present in structure has no need to be examined by an expert, that an image with a ratio above 2% needs to be looked at by an expert.

Since the available data set contains no information regarding how much rust in a structure can be considered too much, a cutoff ratio of 5% was defined for this project, meaning that it should be looked at by an expert if more than 5% of the structure contains rust.

B. SegNet Approach

System Architecture: The high level overview of the system based on utilizing SegNet can be seen on Figure 6, which is very similar to the system architecture of the sliding window architecture described previously. However there are some differences to be taken into account. The pre-processing done is different, while the feature extractor and classification blocks were replaced by a single block that is the SegNet block. The result analysis is also slightly different in each case, since the output of the SegNet block is a prediction mask, while the output of the classification block in Figure 3 is the amount of windows classified as ”rust” and ”structure”.

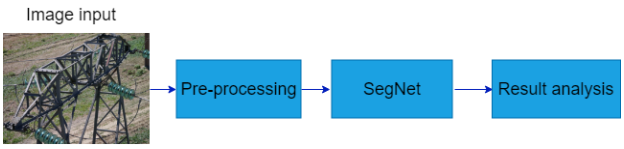


Figure 6: Architecture of the developed system based on a SegNet model

Pre-Processing: The input of a SegNet, like most CNNs, is a static number. As discussed previously, the data set available is composed by images of several different sizes, meaning that, in the pre-processing resize of the training and validation data is necessary.

The size to which the image is resized is an important step, since then larger the image, the larger the amount of memory necessary to train and run the network. However, the larger the image the more detail the network as to work with, which can lead to a better segmentation.

SegNet: Just like implementing a regular CNN, there are several ways of implementing a SegNet model. This can be done in Keras by either creating an encoder and the respective decoder or even by taking an already existing CNN architecture, like VGG [22], take off the fully connected layer, and create the corresponding decoder for this network. However there are some open source libraries that projects the implementation of a SegNet architecture and in this project, a library called *keras_segmentation* [29] was used, which contains several implementations of SegNet architectures, including an architecture based on VGG16.

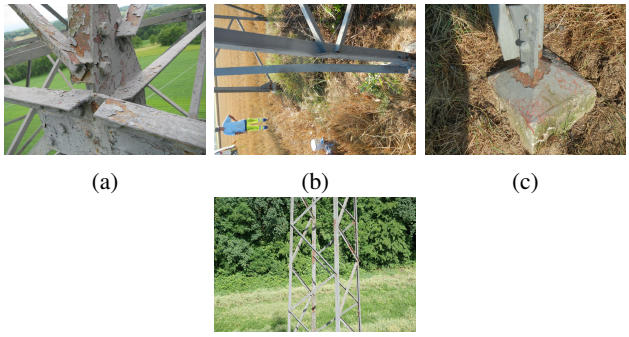
Result Analysis: The way the result analysis is performed for this approach is essentially the same as the way it works for the sliding window approach. However, since the output from the SegNet approach is a segmentation of the original image, in order to obtain the ratio of rust in the structure, it is first necessary to calculate how many pixels were defined as ”rust” and ”structure”.

These values can then be used to calculate the ratio defined in Equation 1. Which, as mentioned, previously is the ratio that will determine if the image should or should be shown to an expert.

V. Experimental Results

A. Sliding window and CNN Results

The images used for testing can be seen in Figure 7. Comparing Figures 8 and 9, it is possible to see how, despite the original data set being the same, a change to the definition of what the network considered ”rust” and ”structure” can have an impact during training.

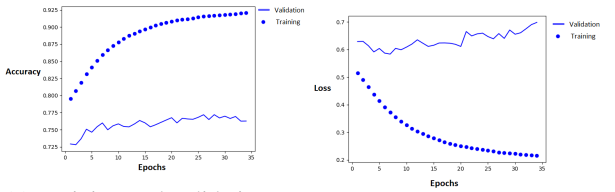


(a) (b) (c)

(d)

Figure 7: Images used to verify the performance of the models (provided by Axians)

However, the results from the training of the network are mainly for understanding how the network is learning and if it is capable of generalizing, if the validation loss and accuracy show low accuracy values it can be assumed that the generated data sets have some inherent problems that do not allow the model to correctly distinguish between ”rust”, ”structure” and ”background”.



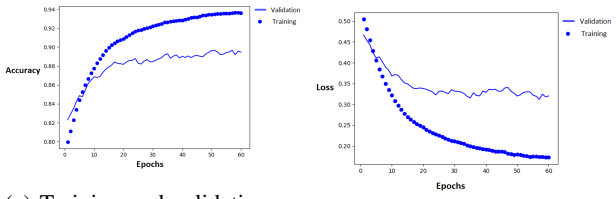
(a) Training and validation accuracy

(b) Training and validation loss

Figure 8: Model trained on 80% ”structure percentage and 75% ”rust” percentage

In Figure 8 it possible to see how the training accuracy tends towards 100% while no improvement occurs on the validation accuracy, which stagnates at about $0,75$. Meanwhile, the training loss decreases at a rapid rate while the validation loss tends to increase. These two signs indicate that the model is not generalizing well and is overfitting.

Looking at the results from training on a different generated data set, Figure 9, it can be seen that the validation accuracy steadily increases until it reaches about 89% accuracy, while the validation loss stagnates at about $0,325$. Although the validation loss does not increase, as seen in Figure 8b, it also does not keep decreasing like the training loss, this means that, even though the training loss reaches a value close to $0,15$, when faced with unknown images the performance of the model will be closer to the validation values, which represent unknown data.

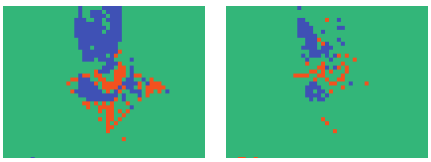


(a) Training and validation accuracy

(b) Training and validation loss

Figure 9: Model trained on 90% ”structure percentage and 90% ”rust” percentage

Segmentation with VGG16: Utilizing the sliding window approach, a total of four images from the original data set were used in order to analyse how the model behaved on a real world setting. In Figure 10, a comparison of the results obtained from training on data sets with two different structure and rust percentage can be seen.



(a) Model trained with 90 ”rust” and 90 ”structure”

(b) Model trained with 25% ”rust” and 40% ”structure”

Figure 10: Comparison of models with same window and stride, but trained on different data sets (provided by Axians)

Table V shows, the average intersection over union values for each percentage of ”rust” and ”structure”. From these results, it was chosen to continue the testing with values for the ”rust” percentage of 90% and ”structure” percentages of 90% and 60%. Also, in Table IV, can be seen the results of the execution time for each image and for different strides.

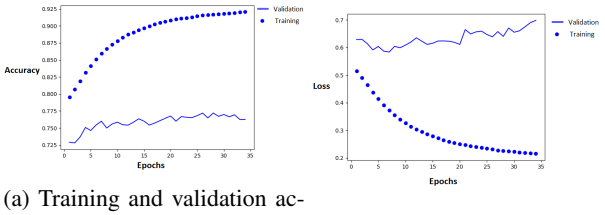
Testing the MobileNet Architecture: The MobileNet architecture, being a less computationally intensive network is expected to have weaker performance when compared to the VGG16. This network was trained on two data sets, both of them with a ”rust percentage” of 90% and with ”structure

	Average detection time (seconds)	
Image Analysed	Stride 64	Stride 32
Figure 7a	25.445	82.7
Figure 7b	99.05	355.64
Figure 7c	17.06	57.09
Figure 7d	69.31	246.87

Table IV: Average detection time of the system using the VGG16 architecture on a *64x64* window

percentage" of 60% and 90%. Since both of these models had similar performance in the both the VGG16 approach and in the MobileNet, the results of the intersection over union and of the execution time is the average of the values obtained resorting to both data sets.

The results during training are, as seen on Figure 11, inferior when compared to the ones obtained by the VGG16. Both the validation accuracy and loss of the model, although they do increase and decrease respectively, they do so in a very inconsistent manner, which shows that the model is not generalizing very well.



(a) Training and validation accuracy

(b) Training and validation loss

Figure 11: Accuracy and loss of the MobileNet, trained on "rust" and "structure" percentage of 90% with *64x64* window

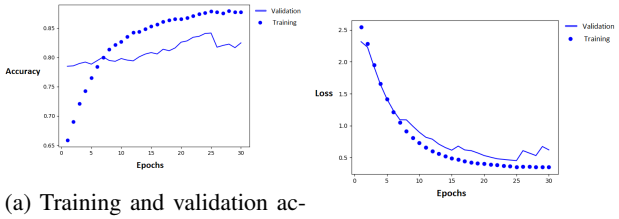
The MobileNet architecture obtained the following results shown in Table VI. From these results, it can be seen that this architecture performed considerably worse than the VGG16.

***Result analysis**:* The results of the ratio of rust existing in the structure obtained from both of the architectures can be seen in Table VII. From these results it is possible to see that there are some cases were the ratio, in both cases, differs from the ground truth by a large margin.

B. SegNet Results

The accuracy and loss curves for the SegNet model trained with an input of *1024x1024* can be seen in Figure 12. It is possible to see that the validation accuracy and loss are somewhat irregular, which to be expected since the VGG16 is a deep neural network that was trained end-to-end on a small data set. Even so the model starts to stagnate at around 80% accuracy with a loss of 0.5, it should be noted that, in the final five epochs, the loss tends to start increasing, which is a sign that the model could be starting to overfit the training data.

Something to take into consideration is that the input of the SegNet must be resized, which may lead to a loss of information and consequently to a larger error on the calculated rust-structure ratio.

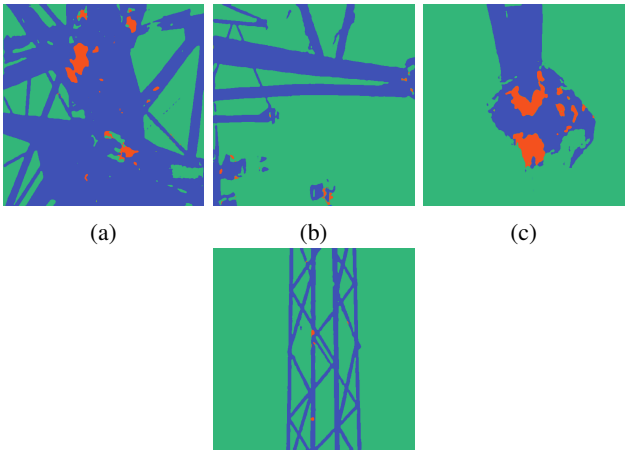


(a) Training and validation accuracy

(b) Training and validation loss

Figure 12: Accuracy and loss for SegNet model trained with images of size *1024x1024*

***Segmentation results**:* Figure 13 shows the segmentation's obtained utilizing this model, while Table VIII show the results of the intersection over union with the original mask, as well as the execution time of this methodology. Looking at these results, it is possible to see that the values of the intersection over union are quite good, given that all of them are above 75%, however, in particular in Figure 13b the shortcomings of the training data set can be seen. Since the training and validation data set have few examples of people in the background, the models decides that the closest thing to the clothes is the class "structure", while determining that the face and hands are more similar to "rust".



(a) (b) (c)

(d)

Figure 13: Segmentation obtained from using a SegNet methodology with an input size of *1024x1024*

***Ratio Calculation**:* In Table IX, the values of the ratio of rusted structure can be seen for the original mask, the resized original masks and the predictions. It is to be noted that, a resize in the image will lead to an error in the percentage of structure that is corroded.

Some values obtained by the SegNet architecture are closer to the ground truth ratios than the resized ground truth image ratios. However this is due to the models not being 100% accurate, which is not ideal. It is also important to note that the resized ground truth in both cases showed a reduction in the value of the rust-structure ratio, which is a disadvantage of using this approach.

	Rust (%)				Structure (%)			
Average intersection over union	25	50	75	90	40	60	80	90
Stride 64	60.53%	60.99%	61.71%	64.08%	62.08%	63.83%	56.96%	64.46%
Stride 32	60.26%	60.74%	61.42%	63.66%	61.87%	63.19%	56.93%	64.09%

Table V: Average of the intersection over union for each value of ”rust” and ”structure” percentage for sliding window with stride 64 and stride 32

Image Analysed	Average intersection over union
Figure 7a	31.70%
Figure 7b	76.25%
Figure 7c	63.37%
Figure 7d	25.22%

Table VI: Average intersection over union of the MobileNet model, using a 64x64 window with stride 64

Image Analysed	Ground Truth	MobileNet	VGG16
Figure 7a	9.55%	0%	13.48%
Figure 7b	2.29%	0%	13.83%
Figure 7c	15.15%	3.53%	34.62%
Figure 7d	6.06%	1.12%	1.52%

Table VII: Percentage of structure that contains rust calculated for the MobileNet and VGG16 architectures

Image Analysed	Intersection over Union	Time (s)
Figure 7a	76.54%	2.73
Figure 7b	82.00%	0.37
Figure 7c	76.89%	0.43
Figure 7d	83.69%	0.36

Table VIII: Intersection over Union and execution time of SegNet segmentation for an input of *1024x1024*

Image Analysed	Ground Truth	Resized	SegNet
Figure 7a	9.55%	8.10%	3%
Figure 7b	2.29%	1.17%	0.84%
Figure 7c	15.15%	10.21%	15.28%
Figure 7d	6.06%	1.6%	0.52%

Table IX: Percentage of structure that contains rust calculated for the SegNet approach.

VI. Conclusions

The goal for this work was to build a system with the goal of being able to detect the amount of rust existing in an image and use that to determine if it needs human inspection. For this two deep learning systems were used, an approach based utilizing a traditional CNN with a sliding window to detect areas with ”rust” and ”structure” and another approached based on utilizing a SegNet to generate a semantic segmentation of the image and using that image to see how many pixels were classified as ”rust” and how many pixels were classified as ”structure”.

The idea of using a sliding window approach came from the results obtained in the work done by Atha [4]. This approach was put into practice with the goal of detecting regions on images that contained rust and implemented a VGG16 architecture, which obtained results with high accuracy for rust detection. Given these high accuracy results, it was chosen to experiment with this approach and create a system that could detect both ”rust” and ”structure”.

The results obtained from applying the sliding window approach were worse than expected, which could have been due to the construction of the data set used in order to train the networks. There are also cases where the approach presents higher than desired execution times, which could be a problem if there is a large quantity of images to evaluate. It was concluded that, with these results, the sliding window system was not capable of detecting rust on structure in a reliable manner.

After obtaining results that were less optimal than the ones expected, it was proposed the implementation of a SegNet approach that could directly output an image segmentation from the original image, which could in turn be used to determine how much of the pixels of an image were classified as ”rust” and how many where classified as ”structure”. When testing this approach it was found that the obtained segmentation was much smoother than a segmentation obtained using a sliding window approach. More importantly, this approach proved to show higher accuracy and faster execution time on the same limited data set. However it does suffer from the need to resize an image before segmenting it, which may affect the ratio between the amount of ”rust” and ”structure”, making it less accurate.

This work shows that it is possible to build a system bases on semantic segmentation that is capable of successfully detect the amount of rust existing in a structure. Although there are some problems involving the change in image size affecting the amount of rust present in a structure, the system was still capable of producing an acceptable segmentation of the the image, given the limited data available, and showed promising results determining the amount of rust present in a structure. Utilizing a SegNet in order to detect the presence of rust in structure seems to be a promising way of approaching rust detection in industrial structures in the future.

VII. Future Work

This work demonstrates that utilizing semantic segmentation approaches is an effective way of detecting the presence of rust and structure in an image. The SegNet model based on the VGG16 showed showed better performance than the sliding

window approach with a limited data set even though it was not pre-trained on any kind of larger data set. Training this model on a large segmentation data set like CityScapes [30] could help improve its performance by later fine-tuning the model with the Axians data set.

An interesting proposition for future projects is to experiment with different kinds of segmentation models, approaches like U-Net [31] and fully convolutional networks [32] are other semantic segmentation methods that have been used and have also performed well in other applications, namely the U-Net architecture as shown impressive results in medical applications [33].

Techniques based on instance segmentation could also be interesting experiments, since it could allow the development of a system that detects distinct rust areas. This makes it possible to develop an approach that analysis each section of the image, or one that presents all rusted areas detected in a structure for evaluation. Architectures like Mask-RCNN [34] are very effective at instance segmentation and are even possible to be used on very high definition videos to detect objects in real time.

References

[1] G. Koch, J. Varney, N. Thompson, O. Moghissi, M. Gould, and J. Payer, "NACE 2016 Impact Study," tech. rep., 2016.

[2] P. Viola and M. Jones, "Managing work role performance: Challenges for twenty-first century organizations and their employees.," *Rapid Object Detection using a Boosted Cascade of Simple Features*, pp. 511–518, 2001.

[3] T. Surasak, I. Takahiro, C. H. Cheng, C. E. Wang, and P. Y. Sheng, "Histogram of oriented gradients for human detection in video," *Proceedings of 2018 5th International Conference on Business and Industrial Research: Smart Technology for Next Generation of Information, Engineering, Business and Social Science, ICBIR 2018*, pp. 172–176, 2018.

[4] D. J. Atha and M. R. Jahanshahi, "Evaluation of deep learning approaches based on convolutional neural networks for corrosion detection," *Structural Health Monitoring*, vol. 17, no. 5, pp. 1110–1128, 2018.

[5] P. Viola, M. Jones, and M. Energy, "Robust Real-Time Face Detection Intro to Face Detection," *International Journal of Computer Vision*, vol. 57, no. 2, pp. 137–154, 2004.

[6] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Region-Based Convolutional Networks for Accurate Object Detection and Segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 1, pp. 142–158, 2016.

[7] V. Badrinarayanan, A. Handa, and R. Cipolla, "SegNet: A Deep Convolutional Encoder-Decoder Architecture for Robust Semantic Pixel-Wise Labelling," 2015.

[8] M. R. Jahanshahi, J. S. Kelly, S. F. Masri, and G. S. Sukhatme, "A survey and evaluation of promising approaches for automatic image-based defect detection of bridge structures," *Structure and Infrastructure Engineering*, vol. 5, no. 6, pp. 455–486, 2009.

[9] M. R. Jahanshahi and S. F. Masri, "Parametric Performance Evaluation of Wavelet-Based Corrosion Detection Algorithms for Condition Assessment of Civil Infrastructure Systems," *Journal of Computing in Civil Engineering*, vol. 27, no. 4, pp. 345–357, 2012.

[10] L. Petricca, T. Moss, G. Figueroa, and S. Broen, "Corrosion Detection Using A.I : A Comparison of Standard Computer Vision Techniques and Deep Learning Model," *Computer Science & Information Technology (CS & IT)*, pp. 91–99, 2016.

[11] T. K. Hanji, T. and K. Kitagawa, "3-D shape measurement of corroded surface by using digital stereography," 2003. Structural Health Monitoring and Intelligent Infrastructure: Proc., 1st Int. Conf. on Structural Health Monitoring and Intelligent Infrastructure, Swets & Zeitlinger B.V., Lisse, The Netherlands, Vol. 1, 699–704.

[12] V. Pakrashi, F. Schoefs, J. B. Memet, and A. O'Connor, "ROC dependent event isolation method for image processing based assessment of corroded harbour structures," *Structure and Infrastructure Engineering*, vol. 6, no. 3, pp. 365–378, 2010.

[13] H. Furuta, T. Deguchi, and M. Kushida, "Neural network analysis of structural damage due to corrosion," tech. rep., 2002.

[14] S. Lee, L. M. Chang, and M. Skibniewski, "Automated recognition of surface defects using digital color image processing," *Automation in Construction*, vol. 15, pp. 540–549, 7 2006.

[15] R. M. Haralick, I. Dinstein, and K. Shanmugam, "Textural Features for Image Classification," *IEEE Transactions on Systems, Man and Cybernetics*, vol. SMC-3, pp. 610–621, 11 1973.

[16] F. N. Medeiros, G. L. Ramalho, M. P. Bento, and L. C. Medeiros, "On the evaluation of texture and color features for nondestructive corrosion detection," *Eurasip Journal on Advances in Signal Processing*, vol. 2010, 2010.

[17] K. Y. Choi and S. S. Kim, "Morphological analysis and classification of types of surface corrosion damage by digital image processing," *Corrosion Science*, vol. 47, pp. 1–15, 1 2005.

[18] X. Xie, "A Review of Recent Advances in Surface Defect Detection using Texture analysis Techniques," *ELCVIA Electronic Letters on Computer Vision and Image Analysis*, vol. 7, no. 3, p. 1, 2008.

[19] S. Ghanta, T. Karp, and S. Lee, "Wavelet domain detection of rust in steel bridge images," *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, pp. 1033–1036, 2011.

[20] G. Antonellis, A. G. Gavras, M. Panagiotou, B. L. Kutter, G. Guerrini, A. C. Sander, and P. J. Fox, "ImageNet Classification with Deep Convolutional Neural Networks," *ImageNet Classification with Deep Convolutional Neural Networks*, p. 9, 2012.

[21] L. Fei-Fei, J. Deng, and K. Li, "ImageNet: Constructing a large-scale image database," *Journal of Vision*, vol. 9, no. 8, pp. 1037–1037, 2010.
[22] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," 9 2014.
[23] F.-F. Li, J. Johnson, and S. Yeung, "Lecture 5: Convolutional Neural Networks," 2017.
[24] R. C. Alex Kendall, Vijay Badrinarayanan, "Segnet."
[25] F. Chollet *et al.*, "Keras." https://keras.io, 2015.
[26] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015. Software available from tensorflow.org.
[27] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in PyTorch," in *NIPS Autodiff Workshop*, 2017.
[28] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. C. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 4510–4520, 2018.
[29] D. Gupta, "image-segmentation-keras." https://github.com/divamgupta/image-segmentation-keras.git, 2017.
[30] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The Cityscapes Dataset for Semantic Urban Scene Understanding," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2016-Decem, pp. 3213–3223, 2016.
[31] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 9351, pp. 234–241, Springer, Cham, 2015.
[32] J. Long, E. Shelhamer, and T. Darrell, "Fully Convolutional Networks for Semantic Segmentation," tech. rep.
[33] S. E. A. Raza, L. Cheung, D. Epstein, S. Pelengaris, M. Khan, and N. M. R. B, "MIMONet : Gland Segmentation Using Neural Network," *Computer Methods and Programs in Biomedicine*, vol. 1, no. d, pp. 698–706, 2017.
[34] K. He, G. Gkioxari, P. Dollar, and R. Girshick, "Mask R-CNN," *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2017-Octob, pp. 2980–2988, 2017.