university of groningen / faculty of science and engineering

Bachelor's thesis — Improving AI-based crack detection on masonry structural surfaces

Bachelor's Project Thesis

Horea Bochis, s4034031, h.a.bochis@student.rug.nl
Supervisors: Dr Dimka Karastoyanova & Dr George Azzopardi

Abstract: The detection of cracks in different masonry surfaces, which represents the highest stock of buildings worldwide, is quite important since mostly this labour is done manually and it is quite expensive and subjective. The purpose of this research is to improve an existing crack detection model by using multiclass prediction with U-Net that has a MobileNetV2 encoder, which has been trained for a period of 50 epochs. U-Net is an architecture for semantic segmentation and MobileNetV2 is a computer vision model open-sourced by Google and designed for training classifiers. In this paper you shall get a detailed view of the whole process and how well it performed.

1 Introduction

The integrity of our cities and buildings is one of the most important concepts for our society and more and more buildings require manual inspection which can prove to be rather costly and ineffective[6, 9]. A lot of old masonry buildings still exist proving that when preserved well, the life cycle of such structures may be significantly extended. Different methods are required in order to make this preservation process more efficient and cost effective. Those inspections could be provided way cheaper with the use of software and image analysis.

A lot of people have taken this into consideration and have started to implement image segmentation models with different architectures such as U-net, MobileNet, FPN-InceptionV3 etc. in order to achieve an automated method that can detect possible issues in those masonry surfaces[1, 5]. The problem with those methods is that they work mainly for isolated brick wall images, while pictures that contain other types of objects such as vegetation, water etc. can influence the output of those models.

In this paper, we are going to discuss an implementation that could fix this issue by using multiclass image segmentation. Due to time constraints and available related work we have decided to go with one technique, more precisely, a U-net model since since this method has been previously used with satisfactory results[1]. Initially, U-net was developed for the segmentation of biomedical images [7, 11] but it performs just as well in other cases. More details will be presented in the Data and Methodology chapter of this thesis. The objective of this thesis is to improve the current state of detection by isolating the brick surface cracks such that other objects encompassed in the image do not influence the result of crack appraisal. After those objects have been isolated from the image, we shall focus mainly on estimating whether this technique has improved the already existing software.

This thesis consists of 5 sections: the Introduction (1), Related work (2), Data and Methodology (3), Results (4) and the Conclusion (5).

2 Related work

As image processing and computer vision is an open field, previous research has been made that is relevant to the research and implementation that we are about to make. In this chapter we are going to discuss the earlier research projects, their results and how they are relevant to our current implementation.

2.1 Provided detection software

The provided software itself, is the research done by Dimitris Dais, Ihsan Engin Bal, Eleni Smyrou and Vasilis Sarhosis. This software (according to the authors) is the first software of such sort which implements deep learning for pixel-level crack detection on masonry surfaces. State of the art CNNs pretrained on ImageNet are examined for their efficacy to classify images from masonry surfaces on patch level with MobileNet obtaining the highest accuracy, that is 95.3%. U-net, a deep FCN(fully Convolutional Network), and FPN, a generic pyramid representation, are combined with different

pretrained convolutional neural networks as the backbone of the encoder part of the network to perform the pixel level crack segmentation. U-net-MobileNet and FPN-InceptionV3 attain the highest F1 score, that is 79.6% [1], and outperform the other networks for crack segmentation used in this study..

In this thesis, their results are going to be compared to what we have obtained after improving their software.

2.2 Review and Comparison of Machine Learning Models for Masonry Wall Crack Detection

In their work "Automatic image-based brick segmentation and crack detection of masonry walls using machine learning", Dimitrios Loverdos, Vasilis Sarhosis have managed to implement several models with different techniques (U-Net, U-Net-SM, LinkNet-SM, FPN-SM, DeepLabV3+) for this sole purpose, the majority of the results having around 95% accuracy [5].

2.3 U-Net

U-Net, a Fully Convolutional Network[4] introduced by Ronneberger et al. in 2015, has since emerged as a gold standard for biomedical image segmentation tasks. Unlike conventional convolutional neural networks that focus primarily on image classification, U-Net is tailored to produce pixel-wise segmentations, effectively mapping regions in an image to corresponding class labels [7].

U-Net's architecture can be visualized as a symmetric expanding path, resembling the shape of a "U". This symmetric design comprises a contracting path (downsampling) followed by an expansive path (upsampling). Crucially, the model also introduces skip connections between mirrored layers in the contracting and expanding paths. These skip connections ensure that the network can use features at multiple scales, allowing for precise localization [10].

Contracting Pathway: The initial segment of the U-Net, often termed the contracting or encoding path, is tasked with extracting contextual information from images. A series of convolutional layers, punctuated by max-pooling layers, model form this path. As the architecture dives deeper into the contracting phase, reduction in the spatial dimensions of the feature maps is observed. Contrarily, the depth, denoted by the number of feature channels, experiences an augmentation. This divided behavior facilitates the architecture's comprehension of the overarching context within the image.

Expansive Pathway: The latter phase, also known as the expansive or decoding path, focuses on reinstating the spatial dimensions to their original magnitude. Achieved through the utilization of transposed convolutions, this phase strives for precise localizations. A noteworthy characteristic of the U-Net's expansive phase is the incorporation of skip connections. These connections seamlessly bridge feature maps from the contracting phase, merging them with those in the expansive phase. Such an approach ensures the preservation and integration of high-resolution details that could potentially be obfuscated during the downsampling process.

The introduction of these skip connections stands as an innovation in the U-Net's design, fundamentally reinforcing its exemplary performance in tasks necessitating pixel-precise predictions. By incorporating features across varied resolutions, the U-Net unites both the semantic essence and the spatial intricacies of the image.

Post its inception, U-Net has not only been the archetype for numerous derivatives but its foundational principles have also found applicability beyond the confines of biomedical imaging. Its efficiency, both in terms of computational resource utilization and the number of parameters, coupled with its unparalleled segmentation accuracy for intricate structures, has certainly established U-Net's stature as a benchmark when it comes to image segmentation methods.

Bellow, in picture 2.1 you can see better how the U-Net architecture looks like.

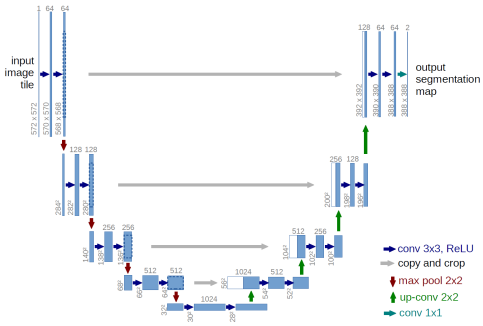

Figure 2.1: U-Net architecture [7]

2.4 MobileNetV2

MobileNetV2, as the name suggests, is the second iteration of the MobileNet architecture, designed with the primary goal of creating a lightweight yet powerful neural network suitable for mobile devices and other edge devices with constrained computational resources [2, 8].

One of the primary innovations introduced in MobileNetV2 is the concept of "inverted residuals." Traditional residual blocks expand the channel dimension, apply convolution, and then project back

to the original channel count. In contrast, MobileNetV2 reverses this process: it first projects to a higher dimension, applies lightweight depthwise separable convolutions, and then projects back using a linear bottleneck. This unique approach helps in reducing computational complexity while retaining representational power. The architecture employs linear bottlenecks to ensure that no non-linearities are introduced in the narrow layers, which could destroy information. These bottlenecks play a crucial role in maintaining the efficiency of the model.

MobileNetV2, like its predecessor, heavily relies on depthwise separable convolutions, a factorized convolution operation that drastically reduces computational cost compared to standard convolutions. By splitting the convolution process into depthwise and pointwise operations, the model achieves efficiency without significant compromises in accuracy. This architecture offers a range of models with different width multipliers, allowing developers to choose a model variant that best fits their specific computational budget and accuracy requirements. In our case, the model captures activations from specific layers of MobileNetV2, similar to the U-Net model.

In figure 2.2, you can find a small schema with more details about MobileNet and MobileNetV2.

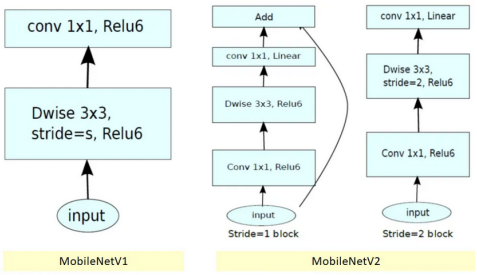

Figure 2.2: MobileNet & MobileNetV2

2.5 Pix2Pix: Image-to-Image Translation with Conditional Adversarial Networks

The "pix2pix" model, proposed by Isola et al. in 2017 [3], is a framework for image-to-image translation using conditional adversarial networks. In essence, it converts types of images into other types, such as turning sketches into colored photos or black and white images into colored ones.

The model utilizes a conditional Generative Adversarial Network (cGAN) approach. The generator follows a modified U-Net structure and attempts to produce outputs that look similar to real images, given an input. The discriminator, in contrast, tries to distinguish between real and fake pairs of input-output images.

Unlike traditional GANs which generate images from random noise, pix2pix requires a paired dataset. That means for every input image, there is a corresponding target output. During training, the generator receives an input image and tries to produce a corresponding output. The discriminator then receives pairs of input-output images and tries to predict if the output is real or generated by the model.

3 Data and Methodology

In this section, we are going to discuss in more detail the whole work process starting from preprocessing the data until the final result. The implementation of the thesis can be accessed at: `https://github.com/HoreaPHP/Crack_detection_2023`.

3.1 Data preprocessing

In order for us to be able to implement this we require a dataset which contains images with masonry surfaces and other types of objects in the background or around the brick surface. We were provided with a dataset consisting of 52 pictures from Amsterdam alongside another dataset that was used for training the model created by Ihsan Engin Bal and his team [1].

After having a look through the data that we have received, we have taken the decision to merge the masks into one since not all of the images had the same number of masks. A part that complicated the process was the fact that the masks were saved in a JPG format. JPG uses a compression algorithm which affects the colors of the pixels, resulting in values that were not consistent in their classes. Considering that the values were not uniform we processed the images into a grayscale format that had constant values and for the best quality we have used a PNG format.

Due to time constraints, we have chosen to stick just with the masks that were related to the cracks themselves since the dataset was rather small and having multiple classes on a dataset like this could prove to be a bit more complicated to optimise, especially when it comes to the weights of the classes. The codebase was built in such a way that it can be easily expanded with more classes.

In figures 3.2 and 3.3, you can find an example of an image and its related mask while in image 3.1 the flow of preprocessing step can be visualized.

Figure 3.1: Code flow

Figure 3.2: Mask

Figure 3.3: Related image

3.2 Neural network

For the neural network, as previously mentioned, we have decided to use a U-Net neural network with MobileNetV2 architecture since MobileNetV2 provides a model that is more lightweight. The main reason why we have chosen to stick to the U-net model was the fact that the previous implementations have also used this model architecture and we wanted to build on that.

As previously stated, due to time constraints, implementing a large variety of techniques would not have been feasible. For the contracting path that we have implemented, we have used a contracting path consisting of 12 layers of 64, 128, 256, 512 respectively 1024 filters. In order to implement the upsampling path in a more convenient way we have used the already implemented usampling path from the pix2pix "tensorflow-example" library. A good way to visualise this architecture would be the previously mentioned figure 2.1.

3.3 Classes and model fitting

Before being fed to the model, the masks were resized from their initial resolution down to a resolution of 1024x1024 alongside their related images. Since the dataset was rather small, the images have also been augmented so that we improve the training process and reduce overfitting by increasing the size of the dataset. By doing this we have managed to increase our number up to 6000 images split in a proportion of 80-20 for training and for validation. After that, 2 classes have been used. Each pixel has been assigned a class based on the color value of it.

The loss function that has been used is categorical-crossentropy alongside the Adam optimiser. Categorical-crossentropy was used since it performs and produces better results in comparison to binary-crossentropy when it comes to multiclass predictions. After combining those, the model has been trained for a period of 50 epochs.

4 Results

In this section, we are going to discuss the results obtained in the methodology section mentioned in the chapters above. It is important to mention that all of the metrics listed below in this section have been tested on the dataset that has been used for training our current model with respect to the model that we were supposed to improve.

4.1 Predictions

In figure 4.1, you can see a couple of predictions resulted from our created model. As you might see the shape of the predicted cracks are the same as the ones from the true mask, just slightly thicker. This thickness might affect the precision metric of the model since some pixel with value 0 are marked as 1.

When you take the two figures 4.1 and 4.2 you can see a clear difference between the two predictions and how we have managed to improve their respective prediction on our dataset.

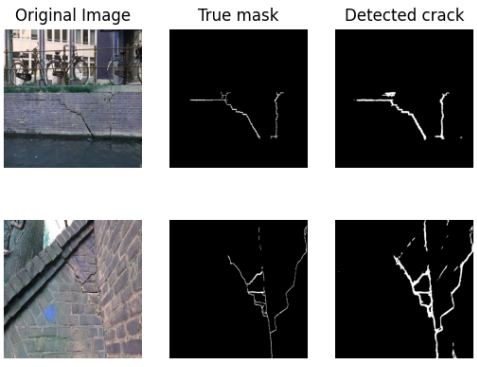

Figure 4.1: 1024x1024 Predictions

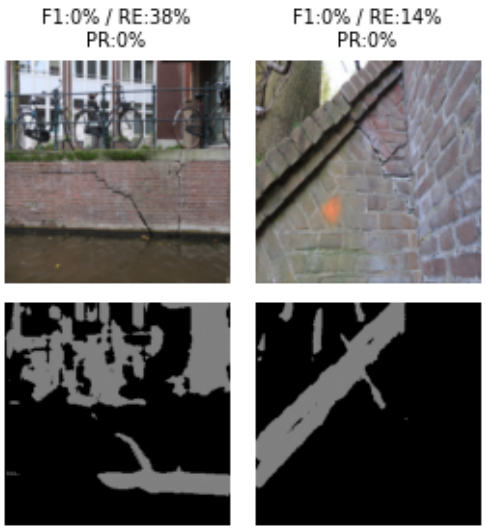

Figure 4.2: 224x224 Predictions

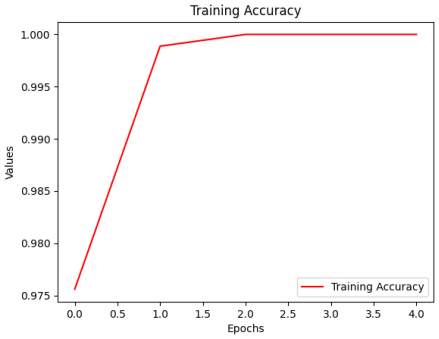

Figure 4.3: Plot of the accuracy

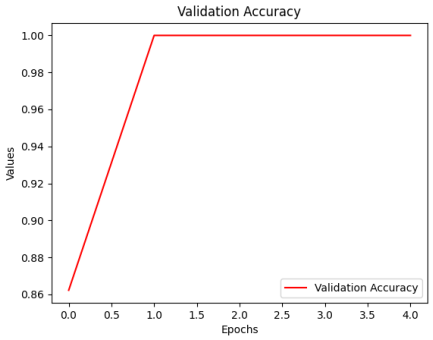

Figure 4.4: Plot of the accuracy for the validation set

4.2 F1 score

The F1 score represents harmonic mean of precision and recall, often used to account for the balance between the two in binary classification tasks. It ranges from 0 to 1, where 1 indicates perfect precision and recall, and 0 indicates neither precision nor recall.

In our implementation, for the 0 class we have obtained an almost perfect F1 score of 1.00. This suggests that the balance between precision and recall for class 0 is excellent. While for the 1 class due to the low precision, we have obtained a score of 0.49. This is a balanced score, but indicates room for improvement.

From the images listed above (4.2) we can see that their F1 score is 0, most probably being resulted from the fact that their precision is equal to 0.

4.3 Accuracy

In the figures 4.3 and 4.4 you can see how the accuracy has increased during the first 5 epochs of the model. The initial improvements in accuracy suggest that the model is learning effectively from the training data. With each passing epoch, the model seems to be refining its predictions, leading to better performance.

4.4 Precision

Precision is the ratio of correctly predicted positive observations to the total predicted positives. It's a measure of how many of the items identified as positive are actually positive. In our case the precision for the 0 class is excellent, having a value of 1.0 while for the 1 class it is about 0.33, proving that the model is still not perfect and there is room for improvement. The precision the we have got is a lot better than the one from the previous implementation.

4.5 Recall

Recall represents the ratio of correctly predicted positive observations to all the actual positives. It's a measure of the classifier's ability to identify all positive instances. In our case we have obtained really good results with 99% for the 0 class and 93% for the 1 class. In comparison to the other software this recall is far greater that it's predecessor.

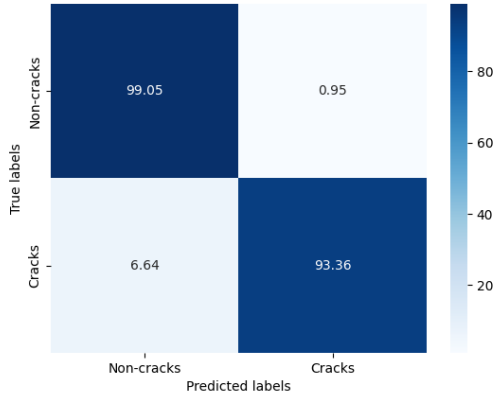

Figure 4.5: Performance matrix

4.6 Confusion matrix

The confusion matrix is a table used to describe the performance of a classification model on a set of data for which the true values are known. The matrix typically contains four values: True Positive (TP), False Positive (FP), True Negative (TN), and False Negative (FN). In our case you can see that our model had a very good prediction in terms of the true-positive values which consist of the correctly predicted pixels.

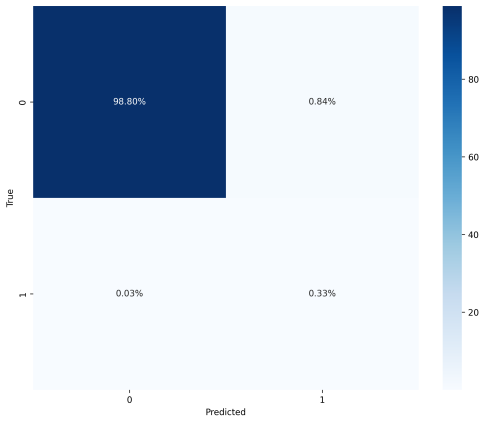

Figure 4.6: Confusion matrix

4.7 Loss function

The loss function is a mathematical function that quantifies the difference between the predicted values and the actual values. It is used during the training of machine learning models to optimize the parameters.

Based on figure 4.7 and 4.8 which plot the loss function over the course of 5 epochs, both the training and validation loss displayed a consistent downward trend, indicative of a model that is effectively learning from its data. Particularly promising is the parallel decline observed in both graphs, suggesting the model's consistent performance on training and validation sets.

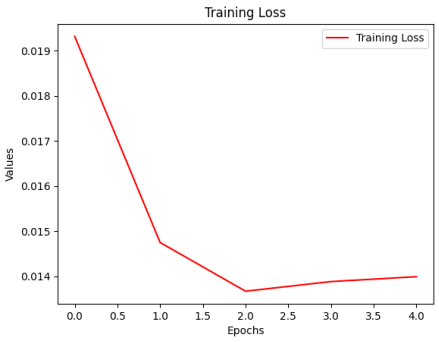

Figure 4.7: Plot of the loss function

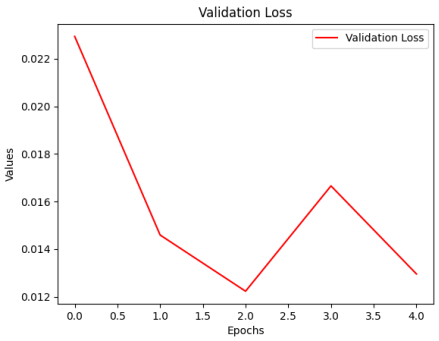

Figure 4.8: Plot of the loss function for the validation set

4.8 Conclusion of the results

The metrics shown in this section mark how well how our model performed in terms of spotting cracks on masonry structural surfaces. The model's ability to identify cracks with few false positives and negatives is demonstrated by its quite high accuracy along with notable precision and recall values. The confusion matrix, alongside the accuracy and loss function plots show more information related to the model's capability to distinguish between objects, demonstrating the dependability of the model.

5 Conclusion

The advancements proposed in this study, particularly the introduction of multiclass prediction, have shown great potential in improving crack detection. The seen results, marked by high accuracy and other performance metrics, attest to the model's potential in automating and improving what has normally been a manual and often subjective task. Furthermore, the model's performance suggests a promising path toward reducing inspection costs and increasing the reliability of structural assessments.

Despite the fact that our model performed well, there is still room for improvement. A couple of things could be done: increasing the variety and size of the native training dataset since the used dataset was rather small and by enlarging it the performance of the prediction can be improved, creating a GUI(Graphics User Interface) in order to improve the quality of the user experience and testing more classes and looking more into the fact that the model still takes some objects into account. The journey toward perfecting AI-driven crack detection is ongoing, but this research marks a significant step forward.

References

[1] Dimitris Dais, İhsan Engin Bal, Eleni Smyrou, and Vasilis Sarhosis. Automatic crack classification and segmentation on masonry surfaces using convolutional neural networks and transfer learning. *Automation in Construction*, 125:103606, 2021. doi:10.1016/j.autcon.2021.103606. URL https://linkinghub.elsevier.com/retrieve/pii/S0926580521000571.

[2] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications, 2017. URL https://arxiv.org/abs/1704.04861.

[3] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks, 2018. URL https://arxiv.org/abs/1611.07004.

[4] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation, 2015. URL https://arxiv.org/abs/1411.4038.

[5] Dimitrios Loverdos and Vasilis Sarhosis. Automatic image-based brick segmentation and crack detection of masonry walls using machine learning. *Automation in Construction*, 140:104389, 2022. ISSN 0926-5805. doi:https://doi.org/10.1016/j.autcon.2022.104389. URL https://www.sciencedirect.com/science/article/pii/S092658052200262X.

[6] Brent M. Phares, Glenn A. Washer, Dennis D. Rolander, Benjamin A. Graybeal, and Mark Moore. Routine highway bridge inspection condition documentation accuracy and reliability. *Journal of Bridge Engineering*, 9(4):403–413, 2004. doi:10.1061/(ASCE)1084-0702(2004)9:4(403). URL https://ascelibrary.org/doi/abs/10.1061/%28ASCE%291084-0702%282004%299%3A4%28403%29.

[7] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. 2015. URL https://arxiv.org/abs/1505.04597.

[8] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks, 2019. URL https://arxiv.org/abs/1801.04381.

[9] A.M. Sowden. *The Maintenance of Brick and Stone Masonry Structures*. E. & F.N. Spon, 1990. ISBN 9780442311667. URL https://books.google.ro/books?id=_2jkSbEaLB4C.

[10] K. Yojana and L. Thillai Rani. Oct layer segmentation using u-net semantic segmentation and resnet34 encoder-decoder. *Measurement: Sensors*, 29:100817, 2023. ISSN 2665-9174. doi:https://doi.org/10.1016/j.measen.2023.100817. URL https://www.sciencedirect.com/science/article/pii/S2665917423001538.

[11] Zongwei Zhou, Md Mahfuzur Rahman Siddiquee, Nima Tajbakhsh, and Jianming Liang. Unet++: A nested u-net architecture for medical image segmentation, 2018. URL https://arxiv.org/abs/1807.10165.