Semantic Segmentation of cracks: data challenges and architecture

Fabio Panella[1,*], Aldo Lipani[1], and Jan Boehm[1]

[1] *Dept. of Civil, Environ. and Geomatic Eng., UCL, London, UK*
[*] *Corresponding author: Fabio Panella, ucesfpa@ucl.ac.uk*

February 3, 2022

Abstract

Deep Learning (DL) semantic image segmentation is a technique used in several fields of research. The present paper analyses semantic crack segmentation as a case study to review the up to date research on semantic segmentation in the presence of fine structures and the effectiveness of established approaches to address the inherent class imbalance issue. The established UNet architecture is tested against networks consisting exclusively of stacked convolution without pooling layers (straight networks), with regard to the resolution of their segmentation results. Dice and Focal losses are also compared against each other to evaluate their effectiveness on highly imbalanced data. With the same aim, dropout and data augmentation approaches are tested, as additional regularizing mechanisms, to address the uneven distribution of the dataset.

The experiments show that the good selection of the loss function has more impact in handling the class imbalance and boosting the detection performance than all the other regularizers with regards to segmentation resolution. Moreover, UNet, the architecture considered as reference, clearly outperforms the networks with no pooling layers both in performance and training time.

The authors argue that UNet architectures, compared to the networks with no pooling layers, achieve high detection performance at a very low cost in terms of training time. Therefore, the authors consider such architecture as the state of the art for semantic segmentation of cracks. On the other hand, once computational cost is not an issue anymore thanks to constant improvements of technology, the application of networks without pooling layers might become attractive again because of their simplicity of and high performance.

KEYWORDS: *semantic segmentation, crack detection, imbalanced data, deep learning, infrastructure monitoring*

1 Introduction

The desire to teach computers how to perform tasks without explicit programming is not new. The birth of Artificial Intelligence (AI) as a discipline is often attributed to the "Dartmouth Summer Research Project on Artificial Intelligence" in 1956. However, 6 years earlier Turing already introduced the concept of Artificial Intelligence with the famous Turing Test [1]. Between 1966 and 1997 AI experienced what is defined as *AI Winter*. It is a sudden interruption of research in this field in part caused by a limit in computing power available at the time. In 1997, the chess-playing computer reopened the way to a seemingly never ending growth of Machine Learning (ML) applications in research as well as industrial projects. Since early 2010s, with the publication of ImageNet [2] and thanks to increasing computational power, Deep Learning (DL) has attracted more research as well as industry applications.

In civil engineering there are many asset management and infrastructure inspection tasks that are potential applications for ML. Focusing on image based visual inspections, there are several DL approaches to perform defect detection such as image classification, object detection and semantic segmentation. The first one consists in recognising the main defect category in the image and classifying the whole image with a single label. Instead, object detection gives the possibility of detecting and locating multiple defects within the same image performing thousands of classifications on the same image. Finally, semantic segmentation represents the last evolution in the progression from coarse to fine inference, providing the user with a per pixel classification of the images starting from a localised classification. The level of automation, the detection accuracy and the information's granularity obtainable with ML techniques is strongly influenced by the employed strategy.

1.1 Motivation

To keep infrastructure in serviceable condition, regular inspections are needed to assess its level of degradation both at structural and material level [3]. To date, the most common approach to perform visual inspection of large infrastructures is still dominated by manual input [4]. In the last decades a large number of researchers have aimed to automate asset management pipelines.

A distinct sub-group of ML-based asset management applications is crack detection. This considers cracks to be one of the first symptoms of structural damage [5, 6, 7, 8, 9, 10, 11]. Early crack detection solutions applying ML are [12, 13, 14]. Since then, the number of proposal for crack detection using ML has grown year by year. The first crack detection attempts are based on the combination of image processing & ML regression tasks [12, 15, 16, 17]. Other researchers address the topic performing crack detection using *DL crack classification* [7, 18, 8, 19, 14, 20] or *semantic segmentation of cracks* [21, 22, 6, 23].

The aim of the present paper is to investigate the applicability of DL to perform semantic segmentation of road pavement cracks. This is a a subject of active research because of the challenge to achieve both high precision and recall. In fact, for civil engineering problems it is very important to detect cracks with a high level of precision, to avoid false alarms, and, at the same time, very high recall scores, to avoid unforeseen incidents. Moreover, datasets with images of cracks are often scarce and characterized by high class imbalance. All of these issues, together with the high interest of the construction industry in automating the inspection process, make this both a demanding and in-demand research area.

1.2 Contributions

The present paper studies semantic segmentation of road cracks. This problem is representative of more general tasks like extracting fine linear structures from cluttered background with semantic segmentation approaches. We contribute to the scientific community by providing key insights to help with the design or selection of effective neural architectures to perform such tasks.

We experimentally verify and quantify the benefits of adopting DL approaches to perform semantic segmentation of cracks against the traditional manual approaches. The analysis, see Table 2 and Figure 4, takes into consideration the segmentation performance as well as the time needed to achieve the final result comparing manual and automated approaches.

We review the effectiveness of the state-of-the-art in solving problems requiring very fine inference as well as learning from highly imbalanced dataset. Multiple versions of UNet [24] and bespoke networks with no pooling layers are trained. The analysis of the segmentation performance on the test dataset allows us to evaluate:

- The effects of pooling layers on the overall segmentation performance: the canonical UNet (UNet64) architecture is compared with the results of the same architecture when the skip-connections are removed (UNet64_NOSC).
- The importance of the model size (in terms of trainable parameters): varying exclusively the number of filters per each block of convolution, three different sizes of UNet are trained (UNet64, UNet32 and UNet16).
- The beneficial impact of a loss function specifically designed for class imbalance (UNet64 VS UNet64_FLxxxx and NoPoolNet-DILC VS NoPoolNet-DILCFL).
- The importance of the receptive field when the pooling layers are removed (NoPoolNet series 2 and 3)
- The benefits of replacing traditional convolution blocks (convolution layers + pooling layers) with dilated convolution operations.

Finally, we evaluate the effectiveness of transfer learning, given the difficulty in retrieving data for civil engineering application (see. 2.2). To do so, the performance in segmenting cracks of the trained models will be tested on new datasets not used during the training process.

1.3 Paper structure

The rest of the paper is structured as follows:

- **Chapter 2**. After an extensive literature review of modern Convolutional Neural Network (CNN) architectures, the data related challenges are investigated. This includes the effectiveness of well known regularization approaches to address them. The chapter will critically review the research on semantic segmentation of cracks.
- **Chapter 3**. We present a set of experiments to study learning in case of imbalanced data and the resolution of inference respectively.
- **Chapter 4**. We give a summary and analysis of the experimental results.
- **Chapter 5**. We conclude the paper arguing our outcomes and highlighting the topics for future research.

2 Literature Review

2.1 CNNs for semantic segmentation

One of the simplest DL architecture is represented by Artificial Neural Networks (ANN). They consist of the serialization of artificial neurons in layers where, at each layer, the input to the i^{th} neuron is forwarded as number and the output is the result of a specific activation function on the weighted sum of the input itself (Equation (1)).

$$x = (x_0, x_1, \cdots, x_N), w = \begin{pmatrix} w_0 \\ \vdots \\ w_N \end{pmatrix} \tag{1a}$$

$$z = x \cdot w + b \tag{1b}$$

$$y = f(z) = \begin{cases} 0, & \text{if } z < t \\ 1, & \text{otherwise} \end{cases} \tag{1c}$$

When it comes to image understanding, the traditional ANN structure has two main limitations. The first limitation is related to the number of parameters (a 28x28 grey scale image has already 784 input parameters) and to the depth of the network potentially exhausting the available computational power. A second limitation is the lack of spatial information, essential in image understanding, since all the neurons are fully connected [25]. To overcome these problems, modern architectures are based on a succession of convolutional and pooling layers. Convolution operations (Figure 1 top) give the possibility to optimize memory consumption and perform agile feature extraction without having to hard code any image processing pipeline. This is achieved thanks to three main properties of convolution operations. *Sparse connectivity* gives the possibility to reduce memory consumption and compute the output with fewer operations using a kernel size smaller than the input size. Doing so, it is possible to detect small and meaningful features storing fewer parameters compared with traditional ANN's [26]. *Parameter sharing* allows the network to learn the response of the input to a specific set of operations rather than having several separate sets of parameters. Finally, the *equivariant representation* property is arguably the reason of the effectiveness of data augmentation. These three properties give CNNs the ability to efficiently handle multidimensional data ($BxHxWxD$ in the case of a B batch of images H high, W wide and with D number of channels) reducing memory requirements and improving statistical efficiency.

Pooling layers (Figure 1 bottom), often used in combination with convolutional layers, can improve the efficiency and the performances of a neural network in several ways. First of all, they reduce the dimensionality of the convolutional output. This translates into reduced computational

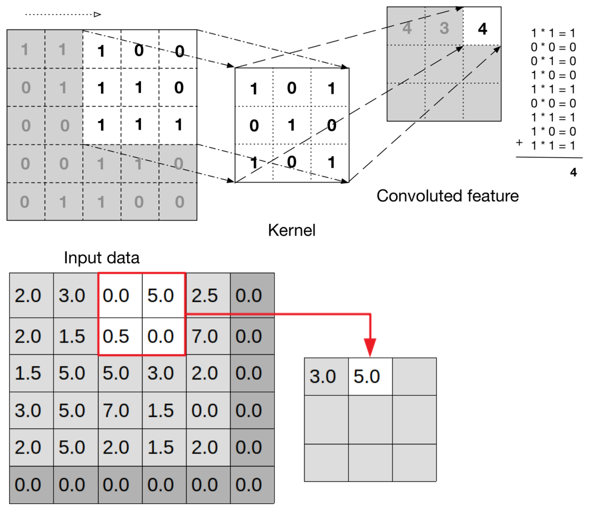


Figure 1: : Example of (top) convolution operation [$k = 3 \times 3$, stride=1] and (bottom) max pooling operation [k=2x2, stride=2].

needs and into better statistical performance. Moreover, it gives the possibility to learn scale invariant features [26].

The networks to perform image semantic segmentation typically consist of two parts, the first is an encoder, often pre-trained, performing feature extraction. This is followed by the decoder performing dense pixel classification by projecting the discriminative features from the encoder to the pixel space. The architectures to perform semantic segmentation are numerous but, in most cases, they are modifications of popular architectures given below:

- Fully Convolutional Networks (FCNs): released in 2015 [27], are usually based on VGG-16 [28], a network previously developed by the Visual Geometry Group for ILSVRC classification (2014). In a departure from the original VGG-16, the final fully connected layers, previously used for the classification task, are removed only leaving convolutional layers, hence the name 'fully convolutional'. The final up-sampling blocks consist of a set of addition and transposed convolutions (FCN_16 and FCN_8) in order to obtain a finer inference compared with the coarse FCN_32's output [27]. This architecture has been implemented in [29]. [9] also use this up-sampling method, but they adopt a 12 layer pre-trained residual network as encoder rather then VGG16. One year later, [21] perform a comparative study analysing the results of multiple architectures as encoding part of FCN_8 architectures.

- U-Net [24], is one of the architectures inspired by FCN [25] and is still one of the most popular for semantic segmentation. Its encoder can based on different architecture types. Unlike FCN, its decoder mirrors the encoder architecture, hence the name of the network. The u-shaped architecture enables a gradual up-sampling granting a finer resolution of the outputs. The architecture proposed in [24] presents skip-connections via concatenating corresponding levels of encoding and decoding blocks. The use of long skip-connections recovers high-resolution spatial information [30], fundamental for semantic segmentation. SegNet [31] can be seen as an evolution of U-Net. Its encoder part is based on VGG-16 and it is connected with its symmetrical decoder with skip-connections. Differently from U-Net, the skip-connections forward to the decoding layers the indices of pooling layers' maximum values to improve the decoding accuracy of the segmentation. [22] implements such architecture for crack segmentation. Another example of U-Net inspired network applied for crack segmentation is [11]. In this paper the loss is calculated at each level of the network to create a multi-scale fusion map optimising the resolution of the pixel-wise prediction.

Above, we described some solutions for performing per pixel classification. Even if they are different from each other, they all share the principle of reducing the spatial dimensionality introducing pooling layers to be able to design networks that are deeper and deeper. A direct consequence of that is the possibility to train a higher number of parameters increasing the non-linearity of the model [32]. However, analysing all the networks for semantic segmentation described so far, it is evident that the low-level features, extracted on the highest resolution, are crucial for improving the performance of the model. Confirming this, [11] argue in their work that all the scales of their network add to the final loss but the higher scales, the ones with same resolution as the input, show heavier contribution to pixel classification robustness. In a similar fashion, the model proposed in [33] evaluates the loss at each scale. However, the novelty of this paper sits in the bottleneck, consisting in a block of dilated convolutions organised both in series and in parallel. Arguably, the authors claim that this helps them to increase the number of features learned. [34] and [35] agree on the idea and add that, if on one side the adoption of pooling layers gives the possibility to increase the number of trainable parameters, on the other side it decreases the resolution of the inference stating that pooling layers might cause loss of information between consecutive convolutional blocks. So, while [34] identify the necessity of residual connections between consecutive convolutional blocks to counteract the loss of information other than to avoid gradients' vanishing, [34] address this issue removing completely the pooling layers.

2.2 Data scarcity and class imbalance

The robustness of a ML algorithm in general, and of CNN architectures in particular, depends on the amount of data fed to the machine to extract information [36, 37, 38]. The need of large training datasets can be reconnected to the *curse of dimensionality* concept introduced in [39] explaining the exponential growth of learning complexity with the linear growth of dimensionality in classification approaches. Especially in civil engineering ML applications, this growth is not reflected by the volume of available datasets which, are often limited by confidentiality issues with the client. One of the possible consequences of small training dataset is overfitting. It means that the statistical model contains more parameters than can be justified by the data. Even if in specific circumstances the overfitting phenomena doesn't negatively affect or even increases the prediction performance of the algorithm [40], it will result in a poor prediction ability of the network in more general cases. Data augmentation [41, 42, 43] is the most common approach to tackle the issue of small datasets. It consists in minor geometric and/or appearance alteration of the images. The benefit is that, applying the same geometric alterations to the target mask, the data volume enlarges without any extra work. Dropout [44] is one of the regularization methods to avoid overfitting. Sometimes, it is also seen as an additional augmentation approach [24]. The value of its hyperparameter ρ, representing the probability of disconnecting neurons at each training step, is usually set to 0.5 for crack detection [45, 18, 21, 8]. Another very popular approach for the same problem is so called transfer learning. It represents the possibility to apply to the target task knowledge leveraged from a source task [46]. Avoiding overtraining with a cross validation approach [26] or avoiding overfitting by undercomputing [47] are two other solutions to face this problem. Finally, a less popular approach to reduce the risk of overfitting is ensemble learning. This approach consists in combining predictions from two or more models with statistics approaches [32, 48, 49]. The drawback of such an approach, especially for semantic segmentation, is the higher computational power required for both training and prediction phases.

Apart from small datasets, another detrimental data related performance issue is *class imbalance* [50, 51, 52]. With this term we refer to the case when the number of samples belonging to one class (minority class) are outnumbered by the samples of another class (majority class). Even if this terminology seems to refer exclusively to a binary classification case, like the one taken into consideration in the present paper, it can be adopted for multi-class classification cases, too [51]. Because of class imbalance, the performance of a trained neural network plummets proportional to the severity of the imbalance [53]. This effect can reduce the accuracy on the minority class up to 10% [50, 54]. There are numerous approaches to perform imbalance learning and they can be classified in techniques acting on data [55], methods applying cost functions to the loss [56] and hybrid approaches [57, 58, 56]. [59] report the effects of class imbalance for DL algorithms explaining why such occurrence causes poor performances of the network. [60] in their review of approaches to address class imbalance categorize the approaches into data, algorithm and hybrid approaches. From the review it emerges that the methods applied for machine learning are extendable to DL with success. However, no conclusive argument is given on which approach is preferable. Moreover, it does not analyse the specific case of semantic segmentation. For semantic segmentation, one of the most common approach for class balancing is to weight the cost function of each class with the inverse of the class frequency multiplied by the median value of all class frequencies [31]. Another possibility is weighting posterior prediction with prior class probabilities at pixel level [61]. While the first approach increases the intersection over union (IoU) to the cost of reduction of the overall accuracy, the second reports noticeable increase in precision and recall.

The combination of data imbalance and small volume of datasets is still a matter of active research [61, 60] and it is a relevant issue in many domains as described in the following section.

2.3 Semantic segmentation of cracks: a case of data scarcity and class imbalance

Given the importance of crack detection for the structural assessment of structures and infrastructure [62, 63], the ideal case scenario would be the highest accuracy on both background and foreground to avoid unnecessary alarms or unforeseen catastrophes. Against such result, data scarcity and class imbalance play an important role.

The severity of class imbalance in crack image datasets can vary and so can the methods to deal with it, depending on which approach of image classification, object detection or semantic segmentation is implemented. For example, consider the case of crack detection based on image classification as performed in [5, 8]. In cases like this, the original class imbalance could be moderate to nil if data sampling is performed correctly. In the case of object detection we have what is referred as *relative imbalance* meaning that the dataset is strongly imbalanced but, increasing the overall dataset volume, the minority class will have sufficient samples to perform fairly well. Instead, in the case of semantic segmentation of cracks, the positive samples (pixels depicting cracks) are consistently heavily outnumbered by the negative ones (background). In this case we can talk of *extreme absolute imbalance*, where the term *absolute* refers to the instance rarity [64] such as the very low absolute number of crack pixels in the image space. [11] test the performance of a weighted loss function to mitigate class imbalance (see Equation 2) against the unweighted one applied to crack segmentation. Several experiments show that the best performance is achieved with $\alpha=\beta=1$ (leading to a total loss function reported in equation 2b) although the class distribution was not balanced. This result contrasts some theories about the weighted loss as an effective approach to mitigate class imbalance. The authors argue their results saying that applying larger weights to the class with smaller occurrence [65, 66] will penalise false negatives resulting in a bigger number of false positives with the potential of undermining the network's performance.

$$l(F_i;W) = \begin{cases} \frac{2\alpha}{\alpha+\beta} \cdot log(1 - P(F_i;W)), & \text{if } y_i{=}0 \\ \frac{2\beta}{\alpha+\beta} \cdot log(P(F_i;W)), & \text{otherwise} \end{cases} \tag{2a}$$

$$L = \sum_{i=1}^{I}(\sum_{k=1}^{K} l(F_i^{(k)} : W) + l(F_i^{fuse}; W)) \tag{2b}$$

with:

F^i: pixel-wise feature map

W: set of standard parameter set

$P(F)$: sigmoid function

α, β: weights of background and foreground, respectively

[8] in a comment related to data scarcity, demonstrate with a parametric study that the minimum number of images needed to obtain reasonable classification performances is 10K. Unfortunately, crack detection is one of the cases where abundance of data is an issue for three main reasons:

- given a surface, whether it is a road pavement or a tunnel lining, we expect the cracked areas to be a small percentage of the total inspected surface.
- in some specific cases like tunnel lining defect detection [5], confidentiality arrangements between the infrastructure's owner and the surveying company may limit the volume of data available, or the possibility to out- or crowd-source the task.
- for crack segmentation a precise per-pixel ground-truth labelling is needed. This manual operation is highly time consuming and very costly if performed by specialised workers [21].

To address this issue, the most adopted approach is data augmentation. [11] and [67] perform an aggressive augmentation consisting of cropping the original image in smaller patches and rotating, flipping and adjusting contrast and brightness the patches. The original datasets, 260 and 250 images respectively, grew to 31,590 and 44,000 images respectively. Finally, [8] noticed the need of samples with different complexity and lighting conditions to obtain a better generalisation.

The benefits of transfer learning applied to address the data scarcity for crack detection, are analysed and quantified in [9, 21]. Instead, [11] demonstrates that a network trained from scratch performs better at crack segmentation compared to the same network initialized with pretrained weights. They argue that natural images, like the Pascal VOC2012 dataset [68], heavily differ from crack images in terms of colours and noise level.

3 Methodology and Experimental Design

The main objective of the experiments is to review the performance of state-of-the-art approaches for the task of semantic segmentation of fine and linear elements with clutter background. The analysis considers both the resolution of the inference and the networks' ability to handle data related challenges.

A first set of experiments consists in training multiple versions of UNet architectures and investigates the effects of pooling layers on the learning process (UNet64 VS UNet64_NOSC), the importance of model sizes (UNet64 VS UNet32 VS UNet16), as well as the impact of the loss function on the learned features (UNet64 VS UNet64_FLxxxxx).

A second set of experiments considers architectures without pooling layers (NoPoolNet). The importance of the receptive field when the pooling layers are removed is quantified (NoPoolNet series 2 and 3). Moreover the benefits of replacing traditional convolution blocks with dilated convolutions is also evaluated.

The comparison between the first set of experiments and the second one gives the possibility to test and in case quantify the benefits of adopting straight architectures (NoPoolNet's) in contrast with the state-of-the-art. Such comparison takes in account not only the performance on the test dataset, but also the transfer learning ability to different datasets. The training time is also considered in the comparison.

3.1 Hyperparameters

To set a baseline we perform a hyperparameter optimization for the UNet architecture. For comparability the best performing set of hyperparameters is applied to the straight network architecture wherever possible. The parameters we analyse are the following:

- Total number of trainable parameters. Due to computational power limitation, the largest straight network has 1.55 M trainable parameters (in the case of 2D convolutions, 5.33 in the case of 2D dilated convolutions). For a fair comparison, two more UNet architectures will be trained other than the canonical one (UNet64): UNet32 (7.48 M) and UNet16 (1.87 M).

- Skip-connections. The benefits of having skip-connections is evaluated comparing UNet64 with its counterpart with no such connections (UNet64_NOSC). The results of the latter architecture will also be used as comparison with the UnPoolNet architectures.

- Dropout. It is a regularizer to avoid overfitting at algorithmic level [44, 26]. It creates a set of sub-networks randomly deactivating connections of the base network with a probability commonly ranging between 0.1 and 0.5. In this way it forces the network to learn robust and concurrent features. In the experiments drop out is applied as per [24] ($\rho = 0.2$ at the last two deconvolution blocks, $\rho = 0.3$ for the classification blocks) or not applied.

- Learning rate. Two initial learning rates are tested: 1E-03 and 1E-04. For all the experiments the initial learning rate is reduced with a factor of 0.1 if the validation loss does not decrease for 8 consecutive epochs (Reduce on Plateau) with a minimum delta of 0.001 (0.0001 for models with Focal Loss), up to a minimum learning rate of 1E-08.

The effectiveness of two different loss functions in mitigating data imbalance is analysed. The first one is the Dice Loss. Introduced by [69] for binary volumetric medical image segmentation it is often used for tasks where high inference resolution is needed [25]. Equation (3) describes the Dice

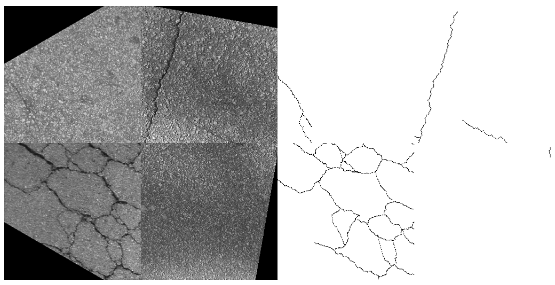


Figure 2: : Mosaic augmentation example: RGB mosaic (left) and related binary labels (right)

loss and highlights the similarities between *Dice Loss* and *Intersection over Union* metric. Such formulation grants learning being less affected by class imbalance. The second is the Sigmoid Focal Loss (see Equation (4)) described in [56] as an effective solution for pronounced class imbalance in dense object detection tasks.

$$D = 1 - \frac{2\sum_{pixels} y_{\text{true}} y_{\text{pred}}}{\sum_{pixels} y^2{}_{\text{true}} + \sum_{pixels} y^2{}_{\text{pred}}} \tag{3}$$

$$CE(p_{\text{t}}) = -\alpha_{\text{t}} log(p_{\text{t}}) \tag{4}$$

For all models *He Normal* [70] is used as kernel initializer of the convolutions, while *Adam Optimizer* [71] is the selected optimizer (with $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 1E - 07$). The maximum number of epochs is set to 300 but the training will stop when the validation loss does not improve more than a Δ_{min} for N consecutive epochs, as it means that the model is fully trained and further iterations would only result in overfitting. This approach takes the name of Early Stopping. Δ_{min} and N are fixed to 0.001 (0.0001 when Focal Loss is used) and 16, respectively.

The metric to evaluate the performance on the test dataset is the average precision calculated as the integration of the precision-recall curve (AUC_PR). Because of its sensitivity, a small difference in the value might represent important variations of the respective curve.

A mini batch of two images is set due to the memory limitation of the GPUs.

3.2 Cracks image datasets

Labelled image data for semantic segmentation of cracks is often limited (see. 2.2). Here we use a publicly available dataset of road cracks (CrackTree260[1], [72]) for training and testing purposes. In this instance, data scarcity is due to the cost of accurate manual labelling. Two different data augmentation approaches are adopted resulting in the same number of images to test the effectiveness of each approach. In the first scenario the augmentation proposed by [11] is applied. Before augmentation a 90% 8% 2% split ratio is performed to divide the dataset in training, validation and test sets respectively. Cracks are identified by one pixel centre line labels. To emphasise class imbalance, no tolerance is considered in the evaluation of the performances both in training and validation phases. Data augmentation consists of nine image rotations (0-90°@10°), vertical and horizontal flipping per each rotated image and 5 cropped patches 512x512 pixels per each flipped/rotated image, one per each corner plus one in the centre [11]. The final number of image patches is 31590, 2700 and 810 for training, validation and test respectively. A different data augmentation scenario consists in implementing the mosaic augmentation approach [73] to the already augmented training and validation data (see Figure 2). For the given dataset, the imbalance for foreground (crack) is 0.5% versus background.

To evaluate the transfer learning ability of the networks, four additional datasets are selected (see section 4).

[1]CrackTree260 dataset and GT: https://1drv.ms/f/s!AittnGm6vRKLyiQUk3ViLu8L9Wzb

3.3 Network architectures

As mentioned above, UNet[2] is the selected network to represent the current research in semantic segmentation. The canonical architecture (*UNet64*) consists of a 4-layer encoder and 4-layer decoder followed by 3 final convolutions representing the classifier. The the last convolution of the classifier has a 1×1 kernel and 2 filters. Each of the convolution blocks in the encoder is a stack of 2 convolutions ($k = 3 \times 3$) each followed by batch normalization and activated with LeakyReLU [74, 70]. After every encoding block, a maxpooling operation is performed, to reduce the spatial dimensionality of data, and the number of convolutional filters is doubled from the initial value of 64 up to a max of 512. After the encoder, an extra convolution block is used as bottleneck. The de-convolution blocks consist of one convolution and one transposed convolution each activated with LeakyReLU after batch normalization. Skip-connections connect the output of every encoding block with a corresponding mirrored block in the decoder. Dropout ($\rho = 0.2$) is applied at each de-convolution block and after the first two convolutions of the classifier block ($\rho = 0.3$). The initial learning rate for this network is 1E-04 and it is reduced as described in the previous section.

The impact of the learning rate is evaluated via the comparison of UNet64 with an identical network trained with higher initial learning rate (*UNet64-03*). In contrast, comparing the canonical architecture with its counterpart without dropout (*UNet64_NODO*) quantifies the impact of the dropout on the overall ability to segment cracks.

Two smaller UNet models (*UNet32* and *UNet16*) are trained. Their architecture is the same as UNet64 but with 32 and 16 initial parameters respectively. The total number of trainable parameters for UNet32 is 7.48 M while UNet16 results in a total of 1.87 M trainable parameters.

UNet64_NOSC's architecture is derived from UNet64 by removing the skip-connections. This exposes the detrimental effects of the pooling operations on inference resolution. It also helps to show the positive effects of skip-connections on preventing gradient vanishing and information loss.

Three more UNet experiments are trained to identify the effectiveness of the Sigmoid Focal Loss. For all networks the same value of $\gamma = 2.0$ is chosen as recommended in [56]. Three different values for the balancing factor alpha have been considered. The first $\alpha = 0.25$ is producing the best results in the reference paper. The values $\alpha = 0.01$ and $\alpha = 0.99$ represent the frequency of foreground pixels and its inverse. These models are named *UNet64_FL02520*, *UNet64_FL00120* and *UNet64_FL09920*.

For the straight networks multiple architectures are proposed to evaluate the effects of model depth, kernel size and type of convolution (dilated or not). All the networks can be divided into 5 subgroups.

- *NoPoolNet-8L128k3* it is a straight network made of 8 identical convolution blocks with 128 filters, each convolution with $kernel\_size = 3$.

- *NoPoolNet2* series has the intent to analyse the effect of having a bigger receptive field for every convolution and demonstrate the importance of context surrounding the pixels of interest for the task of semantic segmentation. Similarly to *NoPoolNet-8L128k3* they are straight networks with constant kernel size and number of filters. Because of the bigger kernel size a the architecture is limited to 6 layers and to a smaller number of filters per convolution. This results in a 70% reduction of the total number of trainable parameters compared with NoPoolNet-8L128k3.

- *NoPoolNet3* series aims to analyse the benefit of reproducing a pyramid effect increasing the kernel size rather than introducing the pooling layers. Starting from a network similar to the *NoPoolNet2* series (6layers and 32 filters) it increases the kernel size of 2 after every second convolution. This also gives the possibility to test deeper networks compared to the *NoPoolNet2* series with kernel size equal to 9 or 11.

- *NoPoolNet4* series resembles the network proposed in [75]. In that paper the authors argue that the number of filters per convolution is less important compared to the network's depth. To validate this theory two similar networks are tested, a deeper one with less filters per convolution (*NoPoolNet4-10L8*: 10 layers with 8 filters per layer, $kernel\_size = 9$) and

[2]The original implementation by [24] is based on caffe. In the present paper we adopt the tensorflow-based implementation from [25] https://github.com/PacktPublishing/Hands-On-Computer-Vision-with-TensorFlow-2/blob/master/Chapter06/unet.py

Table 1: : Overview of network parameters

Model	Train Var.	Loss F	Lr	Dropout	Skipps	K. Size
UNet16	1.87 M	Dice	1E-04	Yes	Yes	3
UNet32	7.48 M	Dice	1E-04	Yes	Yes	3
UNet64	12.73 M	Dice	1E-04	Yes	Yes	3
UNet64-03	12.73 M	Dice	1E-03	Yes	Yes	3
UNet64_NODO	12.73 M	Dice	1E-04	No	Yes	3
UNet32_L1	7.48 M	Dice	1E-04	Yes	Yes	3
UNet32_L2	7.48 M	Dice	1E-04	Yes	Yes	3
UNet32_L1L2	7.48 M	Dice	1E-04	Yes	Yes	3
UNet64_L2	12.73 M	Dice	1E-04	Yes	Yes	3
UNet64_NOSC	11.88 M	Dice	1E-04	Yes	No	3
UNet64_FL00120	12.73 M	Focal	1E-04	Yes	Yes	3
UNet64_FL02520	12.73 M	Focal	1E-04	Yes	Yes	3
UNet64_FL09920	12.73 M	Focal	1E-04	Yes	Yes	3
UNet64_mosaic	12.73 M	Dice	1E-04	Yes	Yes	3
NoPoolNet-8L128k3	1.20M	Dice	1E-04	Yes	Yes	3
NoPoolNet2-6L32k7-03	0.36 M	Dice	1E-03	Yes	Yes	7
NoPoolNet2-6L32k7	0.36 M	Dice	1E-04	Yes	Yes	7
NoPoolNet2-6L32k9	0.59 M	Dice	1E-04	Yes	Yes	9
NoPoolNet2-6L32k11	0.88 M	Dice	1E-04	Yes	Yes	11
NoPoolNet3-6L32k7	0.64 M	Dice	1E-04	Yes	Yes	7++
NoPoolNet3-6L32k9	0.94 M	Dice	1E-04	Yes	Yes	9++
NoPoolNet3NODO-10L32k7plus	1.55 M	Dice	1E-04	No	No	7++
NoPoolNet3-10L32k7plus	1.55 M	Dice	1E-04	Yes	Yes	7++
NoPoolNet4-10L8	0.15 M	Dice	1E-04	Yes	Yes	9,15,36
NoPoolNet4-6L16	0.50 M	Dice	1E-04	Yes	Yes	9,15,36
NoPoolNet_DILC3	4.29 M	Dice	1E-04	Yes	No	3
NoPoolNet_DILC	5.33 M	Dice	1E-04	Yes	No	3, 7, 15
NoPoolNet_DILC_clip	5.33 M	Dice	1E-04	Yes	No	3, 7, 15
NoPoolNet_DILCFL	5.33 M	Focal	1E-04	Yes	No	3, 7, 15

NoPoolNet_DILC_clip has the same architecture as NoPoolNet_DILC but the gradients have been clipped as per [76]
column **K. size**:

- *"++"* indicates the kernel size increases by 2 after each convolution block.
- three numbers indicate a constant kernel size (first number) except for the last two convolution blocks (second and third number).

a shallower one (*NoPoolNet4-6L16*: 6 layers with 16 filters per layer, *kernel_size* = 9). Both of them end with the same classifier consisting of 3 convolution (*kernel_size* = 15, *kernel_size* = 36, *kernel_size* = 1 respectively).

- *NoPoolNet_DILC* series is needed to compare the benefits of dilated convolution (kernel size=3, dilation rate=2 strides=1) against the other networks. *NoPoolNet_DILC3* is obtained from UNet64 by removing the skip-connections and pooling layers and replacing the convolutions with a dilated convolution. *NoPoolNet_DILC* and *NoPoolNet_DILCFL* have larger kernel sizes at first and second convolution of the classifier block to connect each pixel with a larger visual receptive field [75]. *NoPoolNet_DILCFL* replaces the *Dice Loss* with the *Sigmoid Focal Loss*

Table 1 summarizes the main parameters of each network.

Table 2: : Manual labelling baseline

Image	Manual	DL	Recall	Precision	Precision (± 2pix)	AUC_PR
1	6min	0.038sec	72.5%	33.6%	94.4%	24.9%
2	11min	0.040sec				
3	7min	0.040sec				
4	35min	0.039sec				
5	6min	0.038sec				
6	21min	0.041sec				

Performance metrics in this table compare manual labelling performed by the authors against the GT of the benchmark dataset.

4 Results

This section of the paper summarizes the results of the experiments described in 3. It starts with the comparison between manual and automated segmentation of cracks. After that, it continues with the analysis of UNet architectures' results. In this phase we interpret the effects of skip-connection and other hyperparameters on the networks' performance. The effectiveness of different data augmentation strategies addressing data scarcity is also summarized. Finally, we critically analyse the straight architectures' results comparing them with the UNet models' performance.

4.1 Comparison with manual image annotation

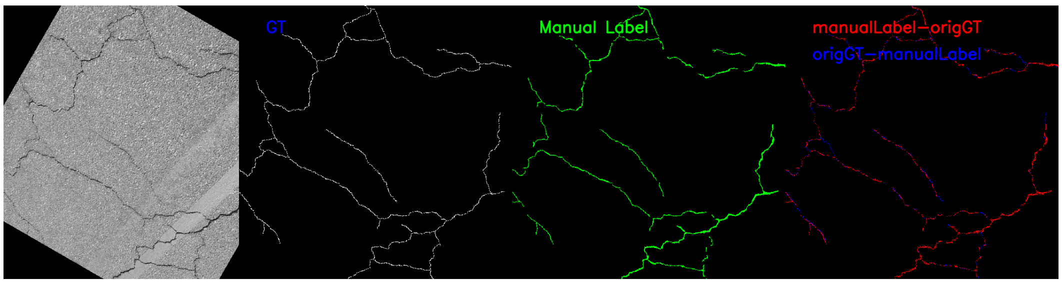


Figure 3: : Comparison between ground truth and manual labelling for the input image (leftmost)

The analysis of the results starts by comparing manual annotations made by the authors to the benchmark's ground truth (GT). This sets the baseline acceptance level for automated segmentation. Comparing time spent on manual labelling with compute time for automated segmentation allows for a cost-benefit analysis. Moreover, this comparison underlines the subjectivity of manual labels and the lack of repeatability of approaches based on manual input (see Figure 3). Considering this subjectivity, we analyse the comparison between manual and automated segmentation of cracks with and without ± 2 pixels tolerance (Figure 4).

If from Table 2 we appreciate the economic benefits, expressed in time spent on the task, of automated detection over the manual labelling, Figure 4 demonstrates how DL semantic segmentation of cracks widely outperforms manual labelling also in the overall performance in finely segmenting cracks. For this analysis, the recall of manual labelling is considered as target and the respective precision is calculated for both manual labelling and DL predictions. The tolerance on the test dataset means that if a False Positive (FP) pixel is within 2 pixels radius from a ground truth (GT) positive pixel it will considered as True Positive (TP).

4.2 UNet performance analysis

In the present paragraph the UNet-inspired models' performance will be interpreted to understand the effects of the skip-connection and hyperparameters. The importance of skip-connections for

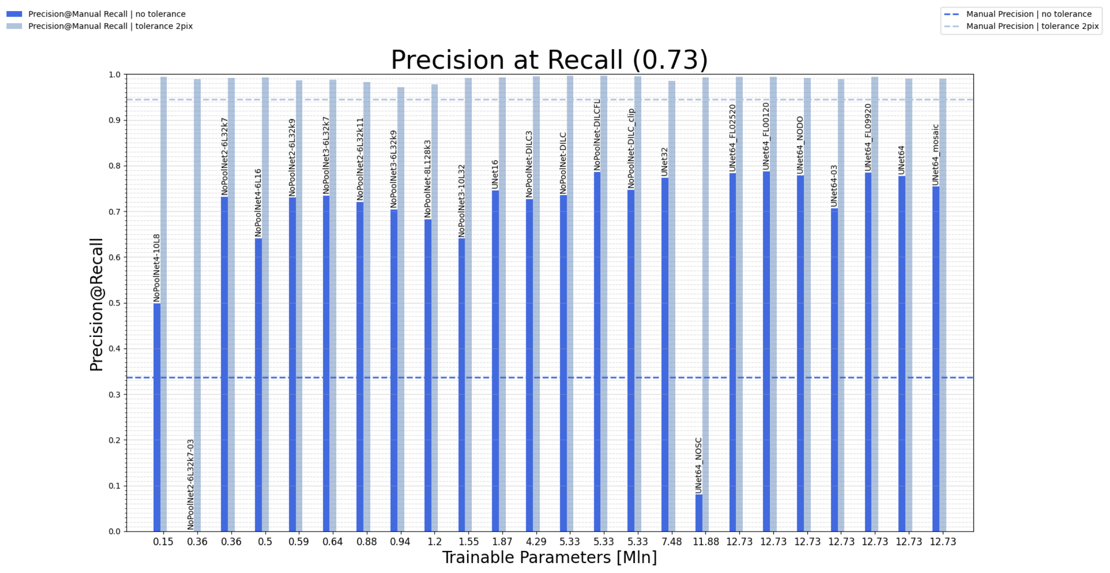


Figure 4: : Precision at manual recall strict and with tolerance.

U-shaped architectures is evident form Figure 4. AUC_PR and $precision@recall(0.73)$ plummet to 64% and 90% respectively when such connections are removed. This seems to validate that pooling layers lead to loss of information, if not supported by skip-connections.

Another hyperparameter strongly impacting the performances of the networks is the initial learning rate for both architecture types. In fact, both U-shaped and straight networks suffer a significant reduction (approximately 14%) of the performances when larger initial learning rate is applied.

The implementation of dropout to address data scarcity is also considered. The difference between the results with (UNet64) and without (Unet64_NODO) dropout layers is barely noticeable and slightly in favour of the case without dropout.

The relation between the number of trainable parameters and the overall performance of the networks is then analysed. The apparent monotonous trend of such relation from Figure 4 is disproved in Figure 5 were UNet16 performs better than UNet32 and almost as well as Unet64 in terms of AUC_PR. This means that having more parameters increases the confidence of the (per-pixel) classification resulting in higher precision when results at lower confidence thresholds are investigated.

Related to data scarcity, the impact of different augmentation techniques is studied. As described in the previous sections, data augmentation performed as per [11] is compared with mosaic data augmentation, both enlarging the original dataset from 260 images to over 30K samples. The results of UNet64_mosaic are compared against UNet64, trained on the dataset obtained with the first augmentation approach. Considering Figure 5, can be noticed that the difference in performance is almost imperceptible. Instead, the difference in $precision@recall(0.73)$ is more evident and in favour of the first approach (Figure 4). The reason for the difference in precision can be researched in the dataset specific characteristics. Mosaic augmentation is designed to make the network *context invariant*, meaning that the networks learns to detect features also in complex scenarios. However, none of the images of the given set of images depicts any content other than cracked tarmac.

Finally, about imbalance learning, from Figure 6 it is evident that, while the validation loss has a positive gradient, the validation AUC_PR is not decreasing. This means that the models are not yet overfitting and suggests that the metric to monitor to activate the early stopping should not be the validation loss but the validation AUC_PR. Such conclusion is confirmed by the networks performance on the test dataset: the models trained with the Focal Loss (UNet64_FLxxxxx) largely outperform (+28%) all the other networks (see Figure 5).

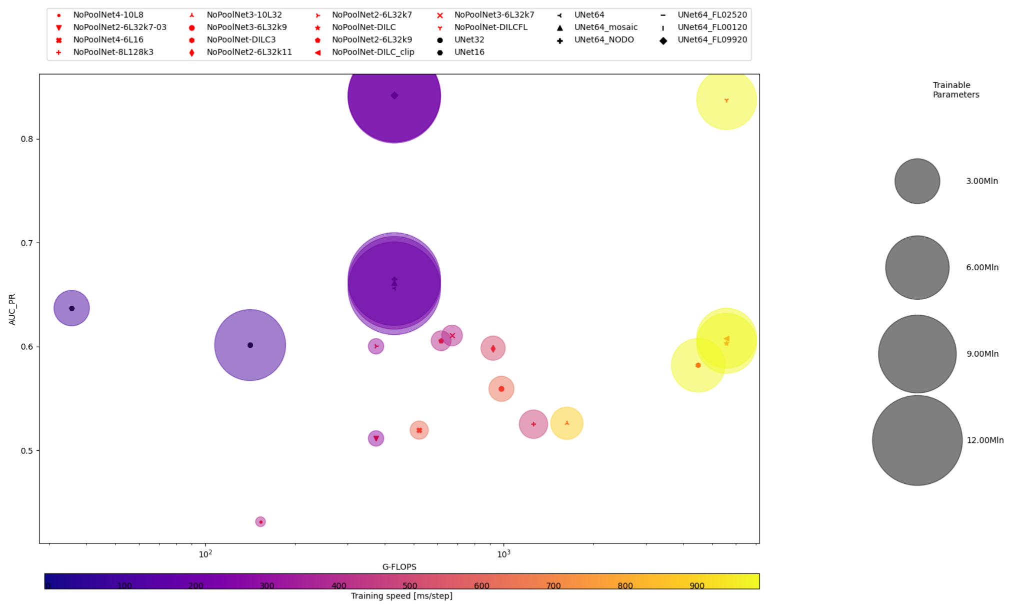


Figure 5: : Multi-parameter analysis.

4.3 UNet versus NoPoolNet

The comparison between UNet and NoPoolNet architectures' performance is in favour of U-shaped ones. However, some further considerations can be made.

As well as the U-shaped architectures, the straight networks suffer of a noticeable performance reduction (approximately 15%) when larger initial learning rate is adopted (NoPoolNet2-6L32k7-03 V NoPoolNet2-6L32k7). With regard to the effectiveness of Sigmoid Focal Loss on handling the class imbalance, similar consideration to the previous paragraph can be made. We noticed a remarkable improvement (+38%) in the segmentation performance when Focal Loss was used even if the validation loss gradient is positive.

Instead, the relation trainable parameters-performance does not follow the same trend as analysed for the UNet architectures. Infact, it can be noticed that the increase in trainable parameters seems counterproductive to the segmentation performance (NoPoolNet2-6L32k7 VS NoPoolNet2-6L32k9 VS NoPoolNet2-6L32k11 and NoPoolnet series 3). However, while in UNet networks the variation of trainable parameters is given by the variation of the number of filters per each convolution, for the straight networks if is related to the dimensions of the receptive field of each convolution. Such difference keeps this topic still open for future research.

For the straight networks, we also test the effectiveness of adaptive clip gradients [76] in avoiding vanishing gradients (NoPoolNet_DILC V NoPoolNet_DILC_clip). The results show a slight advantage when using gradient clipping, justifying its use (Figures 4 and 5 and Table 3).

After the performance analysis on the test dataset, the transfer learning ability of each network is tested. To do so, four more datasets are selected so that they differ from the test dataset (CrackTree260) in terms of lighting, resolution and GT:

- **AIGLE_RN**[3]: includes 38 road pavement images captured at traffic speed using the AIGLE-RN system [77]. Labels are the fusion of the manual segmentation performed by four individuals to counter the lack of repeatability and reliability of human input [78].
- **ESAR**[4]: contains 15 fully annotated images [79].
- **CRKWH100**[5]: consists of 100 road pavement images captured by a line array camera and under visible-light illumination [11].

[3]AIGLE_RN dataset and GT: https://www.irit.fr/~Sylvie.Chambon/Crack_Detection_Database.html
[4]ESAR dataset and GT: https://www.irit.fr/~Sylvie.Chambon/Crack_Detection_Database.html
[5]CRKWH100 dataset: https://1drv.ms/f/s!AittnGm6vRKLtylBkxVXw5arGn6R
CRKWH100 GT: https://1drv.ms/f/s!AittnGm6vRKLglyfiCw_C6BDeFsP

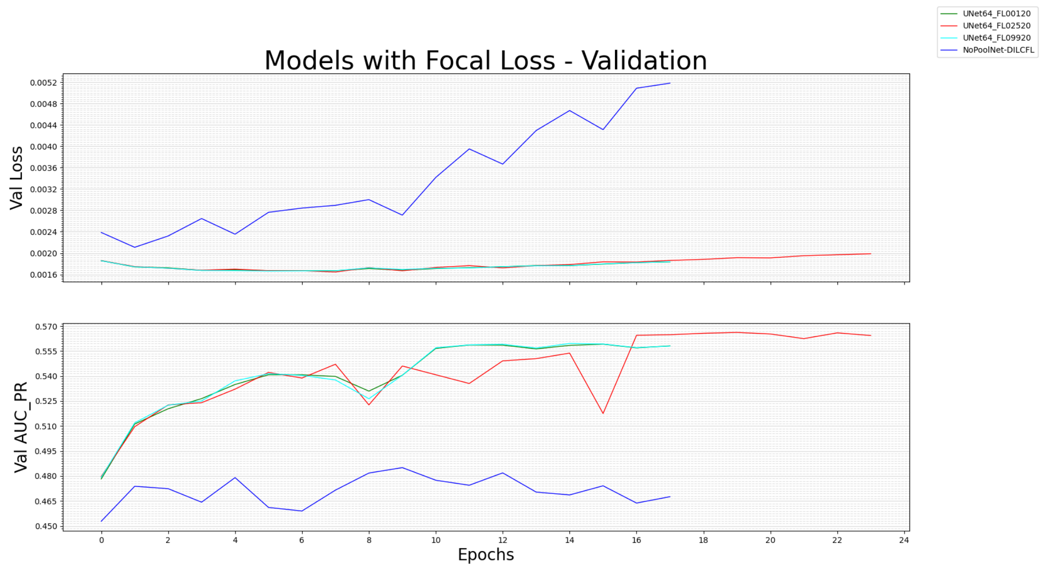


Figure 6: : Models with focal loss: validation performances

Table 3: : Transfer learning performance (AUC_PR). Models trained on CrackTree260.

Label	Train. Var.	CrackTree260	AIGLE_RN	ESAR	CRKWH100	CrackLS3150
UNet64_mosaic	12.73 M	0.6614	0.8882	0.9876	0.6114	0.2279
UNet64_NOSC	11.88 M	0.2360	0.8894	0.9843	0.1215	0.2556
UNet64_NODO	12.73 M	0.6650	0.8884	0.9874	0.6711	0.2265
UNet64_FL09920	12.73 M	**0.8419**	0.9268	0.9814	**0.7933**	0.3730
UNet64_FL02520	12.73 M	0.8406	0.9277	0.9813	0.7865	0.3646
UNet64_FL00120	12.73 M	0.8416	**0.9321**	0.9814	0.7897	0.3794
UNet64-03	12.73 M	0.5668	0.8895	0.9875	0.4819	0.2512
UNet64	12.73 M	0.6561	0.8888	0.9876	0.6214	0.2409
UNet32	7.48 M	0.6014	0.8900	**0.9888**	0.4946	0.2040
UNet16	1.87 M	0.6371	0.8883	0.9875	0.5590	0.2355
NoPoolNet4-6L16	0.50 M	0.5195	0.8903	0.9866	0.3307	0.2604
NoPoolNet4-10L8	0.15 M	0.4313	0.8869	0.9853	0.3359	0.3084
NoPoolNet3-6L32k9	0.94 M	0.5593	0.8895	0.9877	0.5161	0.2261
NoPoolNet3-6L32k7	0.64 M	0.6107	0.8889	0.9874	0.4085	0.2114
NoPoolNet3-10L32	1.55 M	0.5261	0.8884	0.9872	0.5092	0.2624
NoPoolNet2-6L32k9	0.59 M	0.6056	0.8883	0.9874	0.4069	0.2143
NoPoolNet2-6L32k7-03	0.36 M	0.5115	0.8896	0.9879	0.1974	0.2876
NoPoolNet2-6L32k7	0.36 M	0.6002	0.8881	0.9878	0.3073	0.2044
NoPoolNet2-6L32k11	0.88 M	0.5985	0.8885	0.9875	0.4005	0.2484
NoPoolNet-DILC_clip	5.33 M	0.6080	0.8890	0.9877	0.3561	0.2160
NoPoolNet-DILCFL	5.33 M	0.8379	0.9263	0.9814	0.7145	**0.3895**
NoPoolNet-DILC3	4.29 M	0.5820	0.8898	0.9874	0.4344	0.2178
NoPoolNet-DILC	5.33 M	0.6031	0.8892	0.9876	0.2413	0.2092
NoPoolNet-8L128k3	1.20 M	0.5252	0.8894	0.9882	0.2427	0.1986

- **CrackLS315**[6]: consists of 315 road pavement images captured with line array camera but under laser illumination [11].

In Figure 7 the reader can closer inspect the differences between the datasets both in terms of raw images and GT. In the same images are also reported the outputs of the two networks with best performance, one trained with dice loss and the other with the sigmoid focal loss.

Table 3 summarizes the results in terms of AUC_PR without any tolerance per each network and dataset. As expected, the performance on CrackLS315 and CRKWH100 datasets are significantly lower than the one on the test proportion of the CrackTree260 dataset. However, on CRKWH100 and AIGLE_RN datasets, transfer learning produces results unrealistically higher compared with the test dataset. This is explained looking at the respective GTs. While the annotations for CrackTree260, CrackLS315 and CRKWH100 are 1 pixel wide (Figures 7(e) and 7(g)), in the case of AIGLE_RN and Esar the GT looks much wider (Figures 7(f) and 7(h)). In accordance with what noticed for CrackTree260, for all dataset the networks with Focal Loss perform better than the others, with exception of ESAR dataset where UNet32 slightly outperforms UNet64. Interestingly, on CrackLS315 dataset NoPoolNet-DILCFL performs considerably better than all other networks, including UNet64 and UNet64_FLxxxxx.

5 Conclusions and Future Research

In civil engineering, accurate crack detection is important to assist asset managers in their day-to-day job. Information on the exact location, orientation and severity of the cracks is essential for repair and maintenance prioritization of the most critical areas. To that end DL can support the automation of visual inspection. However, even the best networks are currently limited to about 0.8 considering the area under the precision recall curve on the larger datasets (see Table 3 third column). Furthermore, the training and test datasets are still limited in size. Therefore generalization to real world uses cases is still ongoing research. With those limitations in mind, DL algorithms for semantic segmentation of cracks represent a fast, reliable and repeatable approach to automate visual inspection of large infrastructure. As we noticed in chapter 4, deep learning based segmentation outperforms manual labelling both in speed and quality. Moreover, the possibility to transfer learning to similar tasks with decent outcomes has been demonstrated.

From the critical analysis of the results reported in chapter 4, we argue that:

- Data augmentation to enlarge the dataset was successful as the networks were able to learn general features that enabled them to perform the same task on multiple datasets. However, the discussion on which augmentation approach performs better is still open.

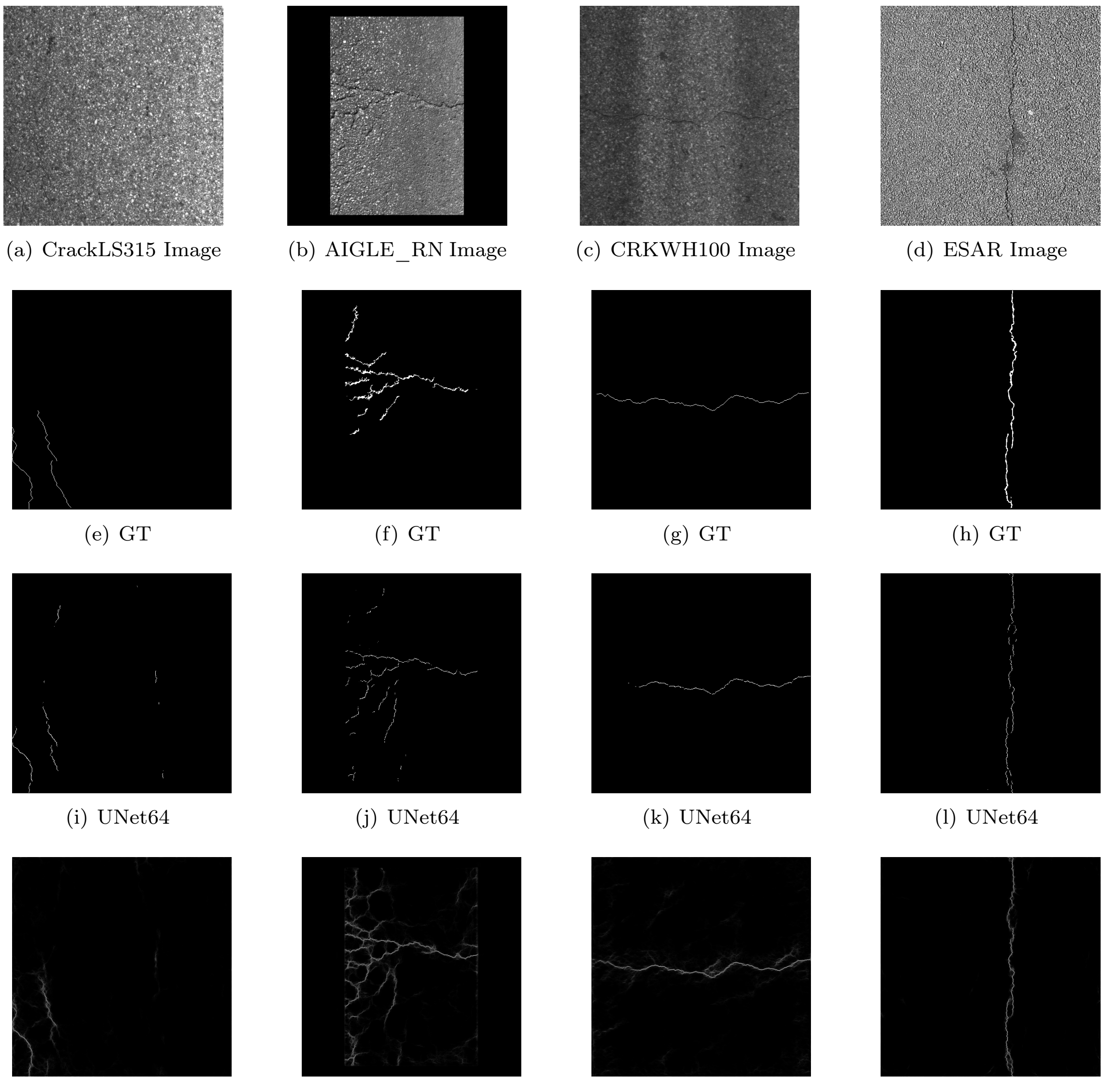
- The detrimental effect of the pooling layers has been confirmed by the analysis of UNet64_NOSC's performances. However, this effect is compensated by the adoption of skip-connection. Such combination results in better performance of the UNet-inspired networks compared with all the straight networks. In the case of straight networks, the gradient clipping approach to avoid gradient vanishing is successful and produces a slight performance improvement (circa 1%) in terms of AUC_PR. However, even with such an approach the straight networks are not able to outperform the UNet-like architectures.

- The expected direct proportionality between the number of trainable parameters and performance (for the same network architecture) has not been confirmed. This might affect future approaches for designing new network architectures especially when training time is a concern. In fact, Figure 5 depicts how UNet16 performs almost as well as UNet64 but with much lower number of G-FLOPS and therefore much faster training. The same has been noticed for the straight networks.

- It is possible to successfully address class imbalance if an appropriate loss function is selected. The improvement in AUC_PR when the focal loss function is applied is noticeable (Figures 4 and 5).

In conclusion, the UNet architecture is considered by the authors as state of the art for semantic segmentation of cracks as it outperforms straight architectures both in performance and training time (see Figure 5).

[6]CrackLS315 dataset: `https://1drv.ms/f/s!AittnGm6vRKLtylBkxVXw5arGn6R`
CrackLS315 GT: `https://1drv.ms/u/s!AittnGm6vRKLg0HrFfJNhP2Ne1L5?e=WYbPvF`


(a) CrackLS315 Image (b) AIGLE_RN Image (c) CRKWH100 Image (d) ESAR Image

(e) GT (f) GT (g) GT (h) GT

(i) UNet64 (j) UNet64 (k) UNet64 (l) UNet64

(m) UNet64_FL09920 (n) UNet64_FL09920 (o) UNet64_FL09920 (p) UNet64_FL09920

Figure 7: Examples of the dataset used for testing purposes only. Examples of the two best models' output are reported, too.

One of the main limitations for the straight networks is computational power. Because of that, it was not possible to train large enough networks or the training time was excessively high. However, it can be said that technology is showing an exponential improvement in terms of computational power. The authors believe that future research on straight networks will provide the tools to produce networks for fine inference. The detailed results of our investigation can be used as a starting point for a targeted hyperparameter optimizations for new application scenarios. We hope that this furthers adopting existing or developing new architectures in the area of semantic segmentation for engineering applications.

References

[1] A. M. Turing, Computing Machinery and Intelligence, Mind LIX (236) (1950) pp. 433–460. doi:10.1093/mind/LIX.236.433.

[2] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, L. Fei-Fei, ImageNet Large Scale Visual Recognition Challenge, International Journal of Computer Vision (IJCV) 115 (3) (2015) pp. 211–252. doi:10.1007/s11263-015-0816-y.

[3] L. McKibbins, R. Elmer, K. Roberts, Tunnels: Inspection, Assessment and Maintenace, no. 671 in CIRIA C, CIRIA, London, 2009, ISBN: 978-0-86017-671-8.

[4] F. Panella, N. Roecklinger, L. Vojnovic, Y. Loo, J. Boehm, Cost-Benefit Analysis of Rail Tunnel Inspection For Photogrammetry and Laser Scanning, ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences XLIII-B2-2020 (2020) pp. 1137–1144. doi:10.5194/isprs-archives-XLIII-B2-2020-1137-2020.

[5] F. Panella, J. Boehm, Y. Loo, A. Kaushik, D. Gonzalez, Deep Learning and Image Processing for Automated Crack Detection and Defect Measurement in Underground Structures, ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences XLII-2 (2018) pp. 829–835. doi:10.5194/isprs-archives-XLII-2-829-2018.

[6] F. Panella, Y. Loo, M. Devriendt, D. Gonzalez, A. Kaushik, R. Ollerhead, J. Boehm, Smart Image Based Technology and Deep Learning for Tunnel Inspection and Asset Management, https://www.researchgate.net/publication/334930778_Smart_Image_Based_Technology_and_Deep_Learning_for_Tunnel_Inspection_and_Asset_Management (Nov. 2018).

[7] K. Makantasis, E. Protopapadakis, A. Doulamis, N. Doulamis, C. Loupos, Deep Convolutional Neural Networks for efficient vision based tunnel inspection, in: 2015 IEEE International Conference on Intelligent Computer Communication and Processing (ICCP), 2015, pp. 335–342. doi:10.1109/ICCP.2015.7312681.

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

[9] S. Bang, S. Park, H. Kim, Y.-s. Yoon, H. Kim, A Deep Residual Network with Transfer Learning for Pixel-level Road Crack Detection, in: 34th International Symposium on Automation and Robotics in Construction, Taipei, Taiwan, 2018. doi:10.22260/ISARC2018/0103.

[10] F.-C. Chen, M. R. Jahanshahi, R.-T. Wu, C. Joffe, A texture-Based Video Processing Methodology Using Bayesian Data Fusion for Autonomous Crack Detection on Metallic Surfaces, Computer-Aided Civil and Infrastructure Engineering 32 (4) (2017) pp. 271–287. doi:10.1111/mice.12256.

[11] Q. Zou, Z. Zhang, Q. Li, X. Qi, Q. Wang, S. Wang, DeepCrack: Learning Hierarchical Convolutional Features for Crack Detection, IEEE transactions on image processing: a publication of the IEEE Signal Processing Society (Oct. 2018). doi:10.1109/TIP.2018.2878966.

[12] M. R. Jahanshahi, S. F. Masri, Adaptive vision-based crack detection using 3D scene reconstruction for condition assessment of structures, Automation in Construction 22 (2012) pp. 567–576. doi:10.1016/j.autcon.2011.11.018.

[13] M. O'Byrne, B. Ghosh, F. Schoefs, V. Pakrashi, Regionally Enhanced Multiphase Segmentation Technique for Damaged Surfaces, Computer-Aided Civil and Infrastructure Engineering 29 (9) (2014) pp. 644–658. doi:10.1111/mice.12098.

[14] D. Soukup, R. Huber-Mörk, Convolutional Neural Networks for Steel Surface Defect Detection from Photometric Stereo Images, in: G. Bebis, R. Boyle, B. Parvin, D. Koracin, R. McMahan, J. Jerald, H. Zhang, S. M. Drucker, C. Kambhamettu, M. El Choubassi, Z. Deng, M. Carlson (Eds.), Advances in Visual Computing, Vol. 8887, Springer International Publishing, Cham, 2014, pp. 668–677. doi:10.1007/978-3-319-14249-4_64.

[15] P. Prasanna, K. J. Dana, N. Gucunski, B. B. Basily, H. M. La, R. S. Lim, H. Parvardeh, Automated Crack Detection on Concrete Bridges, IEEE Transactions on Automation Science and Engineering 13 (2) (2016) pp. 591–599. doi:10.1109/TASE.2014.2354314.

[16] L. Weiguo, L. Yaru, W. Fang, Crack detection based on support vector data description, in: 2017 29th Chinese Control And Decision Conference (CCDC), 2017, pp. 1033–1038. doi:10.1109/CCDC.2017.7978671.

[17] N.-D. Hoang, Q.-L. Nguyen, Automatic Recognition of Asphalt Pavement Cracks Based on Image Processing and Machine Learning Approaches: A Comparative Study on Classifier Performance, Mathematical Problems in Engineering 2018 (2018). doi:10.1155/2018/6290498.

[18] L. Pauly, H. Peel, S. Luo, D. Hogg, R. Fuentes, Deeper Networks for Pavement Crack Detection, in: 34th International Symposium on Automation and Robotics in Construction, Taipei, Taiwan, 2017. doi:10.22260/ISARC2017/0066.

[19] B. Kim, S. Cho, Automated Vision-Based Detection of Cracks on Concrete Surfaces Using a Deep Learning Technique, Sensors 18 (10) (2018) pp. 3452. doi:10.3390/s18103452.

[20] J. Masci, U. Meier, D. Cireşan, J. Schmidhuber, Stacked Convolutional Auto-Encoders for Hierarchical Feature Extraction, in: T. Honkela, W. Duch, M. Girolami, S. Kaski (Eds.), Artificial Neural Networks and Machine Learning – ICANN 2011, Vol. 6791, Springer Berlin Heidelberg, Berlin, Heidelberg, 2011, pp. 52–59. doi:10.1007/978-3-642-21735-7_7.

[21] S. Bang, S. Park, H. Kim, H. Kim, Encoder–decoder network for pixel-level road crack detection in black-box images, Computer-Aided Civil and Infrastructure Engineering 34 (8) (2019) pp. 713–727. doi:10.1111/mice.12440.

[22] J. Ji, L. Wu, Z. Chen, J. Yu, P. Lin, S. Cheng, Automated Pixel-Level Surface Crack Detection Using U-Net, in: M. Kaenampornpan, R. Malaka, D. D. Nguyen, N. Schwind (Eds.), Multi-Disciplinary Trends in Artificial Intelligence, Lecture Notes in Computer Science, Springer International Publishing, 2018, pp. 69–78. doi:10.1007/978-3-030-03014-8_6.

[23] X. Dong, C. J. Taylor, T. F. Cootes, Small Defect Detection Using Convolutional Neural Network Features and Random Forests, in: L. Leal-Taixé, S. Roth (Eds.), Computer Vision – ECCV 2018 Workshops, Vol. 11132, Springer International Publishing, Cham, 2019, pp. 398–412. doi:10.1007/978-3-030-11018-5_35.

[24] O. Ronneberger, P. Fischer, T. Brox, U-Net: Convolutional Networks for Biomedical Image Segmentation, arXiv:1505.04597 [cs] (May 2015). arXiv:1505.04597.

[25] B. Planche, E. Andres, Hands-On Computer Vision with TensorFlow 2, Packt Publishing Ltd, 2019, ISBN: 978-1788830645.

[26] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, MIT Press, 2016, http://www.deeplearningbook.org.

[27] J. Long, E. Shelhamer, T. Darrell, Fully Convolutional Networks for Semantic Segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 3431–3440. doi:10.1109/CVPR.2015.7298965.

[28] K. Simonyan, A. Zisserman, Very Deep Convolutional Networks for Large-Scale Image Recognition, arXiv:1409.1556 [cs] (Apr. 2015). arXiv:1409.1556.

[29] H.-w. Huang, Q.-t. Li, D.-m. Zhang, Deep learning based image recognition for crack and leakage defects of metro shield tunnel, Tunnelling and Underground Space Technology 77 (2018) pp. 166–176. doi:10.1016/j.tust.2018.04.002.

[30] M. Drozdzal, E. Vorontsov, G. Chartrand, S. Kadoury, C. Pal, The Importance of Skip Connections in Biomedical Image Segmentation, arXiv:1608.04117 [cs] (Sep. 2016). arXiv:1608.04117.

[31] V. Badrinarayanan, A. Kendall, R. Cipolla, SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation, arXiv:1511.00561 [cs] (Oct. 2016). arXiv:1511.00561.

[32] G. Wen, Z. Hou, H. Li, D. Li, L. Jiang, E. Xun, Ensemble of Deep Neural Networks with Probability-Based Fusion for Facial Expression Recognition, Cognitive Computation 9 (5) (2017) pp. 597–610. doi:10.1007/s12559-017-9472-6.

[33] Z. Fan, C. Li, Y. Chen, J. Wei, G. Loprencipe, X. Chen, P. Di Mascio, Automatic Crack Detection on Road Pavements Using Encoder-Decoder Architecture, Materials 13 (13) (2020) pp. 1996–1944. doi:10.3390/ma13132960.

[34] N. Lang, K. Schindler, J. D. Wegner, Country-wide high-resolution vegetation height mapping with Sentinel-2, Remote Sensing of Environment 233 (2019) pp. 111347. doi:10.1016/j.rse.2019.111347.

[35] J.-J. Liu, Q. Hou, M.-M. Cheng, J. Feng, J. Jiang, A Simple Pooling-Based Design for Real-Time Salient Object Detection, arXiv:1904.09569 [cs] (Apr. 2019). arXiv:1904.09569.

[36] J. Cho, K. Lee, E. Shin, G. Choy, S. Do, How much data is needed to train a medical image deep learning system to achieve necessary high accuracy?, arXiv:1511.06348 [cs] (Jan. 2016). arXiv:1511.06348.

[37] G. Lemaitre, F. Nogueira, C. K. Aridas, Imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning, arXiv:1609.06570 [cs] (Sep. 2016). arXiv:1609.06570.

[38] K. Bhageshpur, Council Post: Data Is The New Oil – And That's A Good Thing, https://www.forbes.com/sites/forbestechcouncil/2019/11/15/data-is-the-new-oil-and-thats-a-good-thing/, accessed on 01 Jan 2020 (Nov. 2019).

[39] R. Bellman, Dynamic Programming, Science 153 (3731) (1966) pp. 34–37. doi:10.1126/science.153.3731.34.

[40] D. Griffiths, J. Boehm, Rapid Object Detection Systems, Utilising Deep Learning and Unmanned Aerial Systems (UAS) for Civil Engineering Applications, in: ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Vol. XLII-2, Copernicus GmbH, 2018, pp. 391–398. doi:10.5194/isprs-archives-XLII-2-391-2018.

[41] P. Y. Simard, D. Steinkraus, J. C. Platt, Best practices for convolutional neural networks applied to visual document analysis, in: Seventh International Conference on Document Analysis and Recognition, 2003. Proceedings., 2003, pp. 958–963. doi:10.1109/ICDAR.2003.1227801.

[42] B. Raj, Data Augmentation | How to use Deep Learning when you have Limited Data — Part 2, https://medium.com/nanonets/how-to-use-deep-learning-when-you-have-limited-data-part-2-data-augmentation-c26971dc8ced (Apr. 2018).

[43] N. van Noord, E. Postma, Learning scale-variant and scale-invariant features for deep image classification, arXiv:1602.01255 [cs] (Feb. 2016). arXiv:1602.01255.

[44] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: A simple way to prevent neural networks from overfitting, The Journal of Machine Learning Research 15 (1) (2014) pp. 1929–1958. doi:10.5555/2627435.2670313.

[45] Z. Fan, Y. Wu, J. Lu, W. Li, Automatic Pavement Crack Detection Based on Structured Prediction with the Convolutional Neural Network, arXiv:1802.02208 [cs] (Feb. 2018). arXiv:1802.02208.

[46] S. J. Pan, Q. Yang, A Survey on Transfer Learning, IEEE Trans. on Knowl. and Data Eng. 22 (10) (2010) pp. 1345–1359. doi:10.1109/TKDE.2009.191.

[47] I. V. Tetko, D. J. Livingstone, A. I. Luik, Neural network studies, 1. Comparison of overfitting and overtraining, Journal of Chemical Information and Computer Sciences 35 (1995) pp. 826–833. doi:10.1021/ci00027a006.

[48] Z. Fan, C. Li, Y. Chen, P. D. Mascio, X. Chen, G. Zhu, G. Loprencipe, Ensemble of Deep Convolutional Neural Networks for Automatic Pavement Crack Detection and Measurement, Coatings 10 (2) (2020) pp. 152. doi:10.3390/coatings10020152.

[49] A. Kumar, J. Kim, D. Lyndon, M. Fulham, D. Feng, An Ensemble of Fine-Tuned Convolutional Neural Networks for Medical Image Classification, IEEE Journal of Biomedical and Health Informatics 21 (1) (2017) pp. 31–40. doi:10.1109/JBHI.2016.2635663.

[50] N. V. Chawla, K. W. Bowyer, L. O. Hall, W. P. Kegelmeyer, SMOTE: Synthetic Minority Over-sampling Technique, Journal of Artificial Intelligence Research 16 (2002) pp. 321–357. arXiv:1106.1813, doi:10.1613/jair.953.

[51] H. He, E. A. Garcia, Learning from Imbalanced Data, IEEE Transactions on Knowledge and Data Engineering 21 (9) (2009) pp. 1263–1284. doi:10.1109/TKDE.2008.239.

[52] C. Nwankpa, W. Ijomah, A. Gachagan, S. Marshall, Activation Functions: Comparison of trends in Practice and Research for Deep Learning, arXiv:1811.03378 [cs] (Nov. 2018). arXiv:1811.03378.

[53] M. A. Mazurowski, P. A. Habas, J. M. Zurada, J. Y. Lo, J. A. Baker, G. D. Tourassi, Training neural network classifiers for medical decision making: The effects of imbalanced datasets on classification performance, Neural Networks: The Official Journal of the International Neural Network Society 21 (2-3) (2008) pp. 427–436. doi:10.1016/j.neunet.2007.12.031.

[54] K. S. Woods, C. C. Doss, K. W. Bowyer, J. L. Solka, C. E. Priebe, W. P. Kegelmeyer, Comparative evaluation of pattern recognition techniques for detection of microcalcifications in mammography, International Journal of Pattern Recognition and Artificial Intelligence 07 (06) (1993) pp. 1417–1436. doi:10.1142/S0218001493000698.

[55] D. Griffiths, J. Boehm, Weighted Point Cloud Augmentation for Neural Network Training Data Class-Imbalance, in: ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Vol. XLII-2-W13, Copernicus GmbH, 2019, pp. 981–987. doi:10.5194/isprs-archives-XLII-2-W13-981-2019.

[56] T.-Y. Lin, P. Goyal, R. Girshick, K. He, P. Dollár, Focal Loss for Dense Object Detection, arXiv:1708.02002 [cs] (Feb. 2018). arXiv:1708.02002.

[57] A. More, Survey of resampling techniques for improving classification performance in unbalanced datasets, arXiv:1608.06048 [cs, stat] (Aug. 2016). arXiv:1608.06048.

[58] B. Krawczyk, Learning from imbalanced data: Open challenges and future directions, Progress in Artificial Intelligence 5 (4) (2016) pp. 221–232. doi:10.1007/s13748-016-0094-0.

[59] R. Anand, K. Mehrotra, C. Mohan, S. Ranka, An improved algorithm for neural network classification of imbalanced training sets, IEEE Transactions on Neural Networks 4 (6) (1993) pp. 962–969. doi:10.1109/72.286891.

[60] J. M. Johnson, T. M. Khoshgoftaar, Survey on deep learning with class imbalance, Journal of Big Data 6 (1) (2019) pp. 27. doi:10.1186/s40537-019-0192-5.

[61] R. Chan, M. Rottmann, F. Hüger, P. Schlicht, H. Gottschalk, Application of Decision Rules for Handling Class Imbalance in Semantic Segmentation, arXiv:1901.08394 [cs, stat] (Jan. 2019). arXiv:1901.08394.

[62] Comite Euro-International du Beton, CEB FIP Model Code, Thomas Telford, 1990, http://www.tocasa.es/zona2/CEB_FIP_model_code_1990_ing.pdf, accessed on 20 Mar 2020.

[63] The European Union Per Regulation, EN 1992-1-1, BSI Standards Limited, 2004, ISBN: 978 0 580 83726 5.

[64] G. M. Weiss, Mining with rarity: A unifying framework, ACM SIGKDD Explorations Newsletter 6 (1) (2004) pp. 7–19. doi:10.1145/1007730.1007734.

[65] S. Xie, Z. Tu, Holistically-Nested Edge Detection, arXiv:1504.06375 [cs] (Apr. 2015). arXiv:1504.06375.

[66] Y. Liu, M.-M. Cheng, X. Hu, K. Wang, X. Bai, Richer Convolutional Features for Edge Detection, IEEE Transactions on Pattern Analysis and Machine Intelligence 41 (8) (2018) pp. 1939–1946. doi:10.1109/TPAMI.2018.2878849.

[67] B. Wang, W. Zhao, P. Gao, Y. Zhang, Z. Wang, Crack Damage Detection Method via Multiple Visual Features and Efficient Multi-Task Learning Model, Sensors 18 (6) (2018) pp. 1796. doi:10.3390/s18061796.

[68] M. Everingham, L. Van~Gool, C. K. I. Williams, J. Winn, A. Zisserman, Visual Object Classes Challenge 2012 Dataset (VOC2012), http://host.robots.ox.ac.uk/pascal/VOC/voc2012/ (2012).

[69] F. Milletari, N. Navab, S.-A. Ahmadi, V-Net: Fully Convolutional Neural Networks for Volumetric Medical Image Segmentation, arXiv:1606.04797 [cs] (Jun. 2016). arXiv:1606.04797.

[70] K. He, X. Zhang, S. Ren, J. Sun, Delving deep into rectifiers: Surpassing human-level performance on imagenet classification, in: 2015 IEEE International Conference on Computer Vision (ICCV), 2015, pp. 1026–1034. doi:10.1109/ICCV.2015.123.

[71] D. P. Kingma, J. Ba, Adam: A Method for Stochastic Optimization, arXiv:1412.6980 [cs] (Jan. 2017). arXiv:1412.6980.

[72] Q. Zou, Y. Cao, Q. Li, Q. Mao, S. Wang, Cracktree: Automatic crack detection from pavement images, Pattern Recognition Letters 33 (3) (2012) pp. 227–238. doi:10.1016/j.patrec.2011.11.004.

[73] A. Bochkovskiy, C.-Y. Wang, H.-Y. M. Liao, YOLOv4: Optimal Speed and Accuracy of Object Detection, arXiv:2004.10934 [cs, eess] (Apr. 2020). arXiv:2004.10934.

[74] A. L. Maas, A. Y. Hannun, A. Y. Ng, Rectifier nonlinearities improve neural network acoustic models, in: in ICML Workshop on Deep Learning for Audio, Speech and Language Processing, Vol. 30, 2013, https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.693.1422&rep=rep1&type=pdf.

[75] Y. Fei, K. C. P. Wang, A. Zhang, C. Chen, J. Q. Li, Y. Liu, G. Yang, B. Li, Pixel-Level Cracking Detection on 3D Asphalt Pavement Images Through Deep-Learning- Based CrackNet-V, IEEE Transactions on Intelligent Transportation Systems 21 (1) (2020) pp. 273–284. doi:10.1109/TITS.2019.2891167.

[76] A. Brock, S. De, S. L. Smith, K. Simonyan, High-Performance Large-Scale Image Recognition Without Normalization, arXiv:2102.06171 [cs, stat] (Feb. 2021). arXiv:2102.06171.

[77] R. Amhaz, S. Chambon, J. Idier, V. Baltazart, Automatic Crack Detection on Two-Dimensional Pavement Images: An Algorithm Based on Minimal Path Selection, IEEE Transactions on Intelligent Transportation Systems 17 (10) (2016) pp. 2718–2729. doi:10.1109/TITS.2015.2477675.

[78] S. Chambon, J.-M. Moliard, Automatic Road Pavement Assessment with Image Processing: Review and Comparison, International Journal of Geophysics 2011 (2011) pp. 1–20. doi:10.1155/2011/989354.

[79] F. Yang, L. Zhang, S. Yu, D. Prokhorov, X. Mei, H. Ling, Feature Pyramid and Hierarchical Boosting Network for Pavement Crack Detection, arXiv:1901.06340 [cs] (Jan. 2019). arXiv:1901.06340.