Universitat Politècnica de Catalunya

Master in Artificial Intelligence

Facultat d'Informàtica de Barcelona

Bridge Structural Damage Segmentation Using Fully Convolutional Networks

Author:
Juanjo Rubio Guillamón

Supervisor:
Sergio Escalera Guerrero

April 16, 2018

Contents

Abstract

Structural Health Monitoring (SHM) has benefited recently from computer vision, more specifically Deep Learning approaches. In this work we test Fully Convolutional Networks in a dataset of deck areas of bridges in the wild in Japan for delamination and rebar exposure damage segmentation. The dataset has been collected from inspection records and a set of 734 images with 3 labels per image has been obtained, becoming the largest dataset of images in the wild of bridge deck damage. This scenario allows us to estimate the performance of our method based on regions of agreement, emulating the uncertainty in on-field inspections. In this work we prove the capabilities of Fully Convolutional Networks to perform an automated semantic segmentation for detection of surface damages.

El monitoreo de la integridad de estructuras se ha beneficiado recientemente de técnicas de visión por computador, más específicamente de técnicas de *Deep Learning*. En este trabajo ponemos a prueba las redes *Fully Convolutionall* en un dataset generado a partir de registros de inspecciones, el cual está compuesto por 734 imagenes con 3 etiquetas por imagen, convirtiéndose en el dataset de mayor tamaño en entornos no controlados para imagenes de la zona de la cubierta de puentes. Este escenario nos permite estimar el comportamiento de nuestro método basado en el nivel de acuerdo en el etiquetado, emulando la incertidumbre existente en inspecciones. En este trabajo probamos las capacidades de redes *Fully Convolutional* para la tarea de segmentación automatica para la deteccion de daños en superficie.

El monitoratge de la integritat d'estructures s'ha beneficiat recentment de tècniques de visió per computador, mes específicament de tècniques de *Deep Lerning*. En aquest treball posem a prova les xarxes *Fully Convolutional* en un dataset generat a partir de registres de inspeccions, el qual està format per 734 imatges amb 3 etiquetes per imatge, convertint-se en el dataset de major tamany en entorns no controlats per a imatges de la zona de la coberta de ponts. Aquest escenari ens permet estimar el comportament del nostre mètode basat en el nivell d'acord en l'etiquetat, emulant la incertidumbre existent en inspeccions. En aquest traball provem les capacitats de xarxes *Fully Convolutional* per a la tasca de segmentació automàtica per a la detecció de danys en superfície.

1 Introduction

The structural integrity of critical infrastructures need to be evaluated regularly to guarantee their operation safety and to initiate maintenance if required in a timely manner. Historically the process of standard inspection required trained inspectors on-site to assess the health of such infrastructure. The next natural step in structural health monitoring (SHM) was the deployment of various sensors for monitoring physical features for aiding and instructing routine inspection [35] [25]. For some type of monitoring such as internal structure integrity, sensor-based approaches are the only feasible and practical methodology. On the other hand, certain types of damage, specifically those related to the surface of the structure, can be tracked and monitored using image-based approaches. With the advance in optic devices and computer vision, image and video-based approaches to SHM has gained attention from the civil engineering community [34].

Japan transportation infrastructure was mainly built after the World War II and due to its topology present a high number of bridges, rated at 650.000 bridges by the Ministry of Land in 2015 [23]. The severe and extreme climate conditions and frequent natural disasters such as earthquakes and typhoons, SHM has attracted interest of both industry and research [10] for monitoring an aging infrastructure.

In our work, we work with Nagai Lab at Tokyo University [24] to develop a proof of concept for damage detection in the surface of concrete bridges in Japan. Currently, this particular standard inspection is done only via on-site assessment from authorized civil engineers. These inspections have been documented and stored for years and each prefecture has a record of the bridges corresponding to its jurisdiction with manual human evaluation of the image. At present, this evaluation by law has to be performed by an authorized civil engineer and the official guidelines of [23] are defined loosely, yielding a very subjective evaluation with significant variability among different inspectors. This project aims to provide an automated segmentation framework for finding types of surface damage in concrete bridges which potentially can help reevaluate the criteria of damage definition. Another application that was discussed was the use of this methodology in developing countries in which the lack of skilled labor with qualification for infrastructure inspection can benefit from an automated remote evaluation framework.

In order to evaluate the segmentation method, a dataset of 734 images has been collected with multiple labels per sample in order to quantify the agreement among different inspectors. For this proof of concept, the dataset is limited to a certain region of a standard concrete bridge and dealing with the main damages of this part of the structure.

2 Related Works

Traditional approaches for SHM mainly in internal structure evaluation rely on sensor data for measuring physical data in order to quantify and evaluate the health of a particular infrastructure. These methods are based on integrating sensors for analyzing vibration signals [26] [30] [15], measurement of displacement of the structure [13] [9], and in many scenarios, due to legislation or technology limitation the assessment is done on-site via visual inspection.

Computer vision based techniques have been widely used in recent years in applications for surface damage detection. Classic computer vision methods such as thresholding and the use of filters for crack detection in concrete or steel [27] [20] [36] [38]. Ellenberg et al. [8] combined image and infrared imagery from unmanned aerial vehicles (UAVs) to detect delamination on bridges. One of the main drawbacks of these classic approaches is the amount of feature engineering and manual fine tuning for a very specific task with the consequent difficulty of generalizing for other types of damage. Recently, computer vision algorithms have been combined with machine learning such as SVM, or self organizing maps for crack detection [31] [18] [4], and for classifying multiple types of damage in concrete and steel surfaces [19] [28] .

Recently, due to the success of Deep Learning in computer vision tasks, Convolutional Neural Networks (CNN) have been applied to damage detection. Cha et al. [3] applied AlexNet [14] network for cracks on concrete surfaces in a patch-wise analysis of images taken in a controlled scenario. In [3] Faster-RCNN was used for bounding box detection of multiple types of damage of corrosion, delamination and cracks. Zhang et al. [39] developed CrackNet to detect cracks in 3D images of asphalt surfaces.

Thanks to the advances on Deep Learning and in particular by Fully Convolutional Networks (FCNs) for image segmentation, in this work we aim to apply FCN for segmentation on a dataset of texture-based damages on images in the wild, in comparison to related work done on controlled imagery.

3 Dataset

One of the main contributions of this project is the design of a dataset of structural damages suitable for benchmarking texture-based segmentation methods. A detailed review and description of the dataset can be found in the following sections.

3.1 Dataset Description

The dataset consists of images taken from standard infrastructure inspection records of 30 municipalities from the Niigata prefecture in Japan. The images are obtained from the deck area (figure 1) of 9344 bridges yielding an initial set of 2000 images containing damage. After automatic filtering based on resolution and manual inspection from the Nagai Laboratory [24] from Tokyo University based on data quality, an initial set of 734 images have been labeled. This number of images will increase and potentially be available online for research purposes.

The origin of the dataset was purely targeted to documenting visual inspections in standard procedures, which yields to a very rich dataset in terms of real world representation in comparison to datasets created in a controlled scenario. The images, taken by different inspectors present a wide variety of angles, distances and lighting conditions.

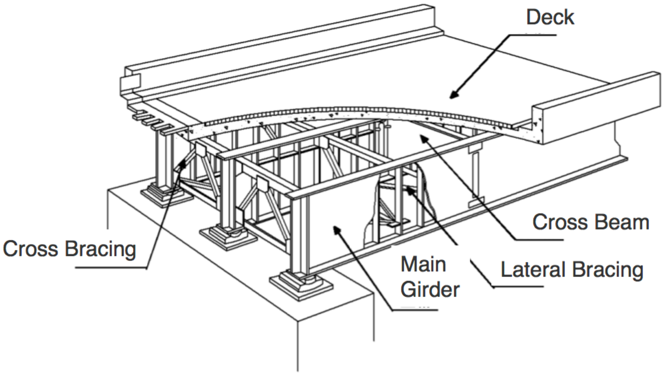


Figure 1: Structure of standard bridge. The dataset is based on the region marked as deck.

In the region of interest there are two main types of damage. Delamination is present when the concrete surface gets detached from the structure,

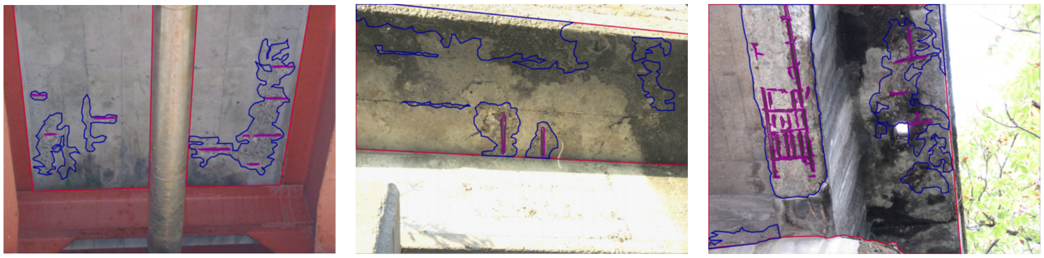


Figure 2: Sample images containing the ROI (red), delamination damage (blue), rebar exposure (purple) and non-damage regions.

mainly due to poor quality of material, building process or due to age of the structure. When delamination takes place, the aggregate of the concrete becomes visible. Heavy delaminated regions yields to the second type of damage, rebar (reinforced steel bar) exposure. Rebar exposure is present when the steel bar is visible and gets corroded due to salt and humidity. Sample images can be seen in figure 2

The images have been categorized into four different levels of severity by authorized inspectors following the guidelines of the Ministry of Land, Infrastructure, Transport and Tourism from Japan [23] regarding the deck area. As of the definition of these types of damage, the official description is as follows:

- **Type A**: No presence of damage.
- **Type C**: Only delamination is observed, including partial damage.
- **Type D**: Rebar is exposed to the air but the corrosion is not severe, including partial damage.
- **Type E**: Rebar is exposed to the air and the corrosion is considerable, including partial damage.

Even though there is an official definition of the levels of damage, the judgement is very subjective to each inspector and the classification in the records database are vague and many images could be interchangeable among different types of damage.

3.2 Dataset Analysis

The dataset has been annotated using the web-based tool LabelMe [32] by 3 civil engineers researchers from the Nagai Lab [24] and 3 external labelers for a subset of data that was outsourced. The external labelers have been trained and instructed by Nagai Lab in order to follow the criteria established. A total of 6 labelers annotated 3 labels per image in order to attempt to model the uncertainty in the process of evaluating the damages in a given inspection image.

The 734 images labeled yield a total of 2202 labels. The split based on the levels of damage described in section 3 is shown in table 1 .

Severity	**# of images**
Type A	4
Type C	185
Type D	510
Type E	35

Table 1: Distribution of images based on damage severity

Even though the dataset seems unbalanced an analysis with Nagai Lab was performed and it was concluded that many images of type D could be interchangeable with Type C and E and for segmentation purposes the textures remain consistent. Practically most of the severity was judged by the inspectors based on size and depth of the damage and type D remains in between very severe and mildly severe, making most of the decisions fall into this category. On the other hand, type A images are very low in number but in terms of effective pixels of non-damage area they represent the majority of the images all across the severity types.

As we can see in figure 3, delamination represents the majority of blobs and blob sizes. The histogram is slightly more spread which represents that sizes and shapes are very diverse, being in general much bigger than rebar exposure blobs. Regarding rebar exposure blobs, in general the labeled regions are much smaller in size and usually present a more constant structure, presenting an elongated thin shape due to being steel bars exposed. The texture representing rebar exposure tends to be more consistent compared to delamination areas since the corrosion on metal is a texture with less variation than the different levels of concrete delamination. Finally, the class non-damage is generally represented as one only blob since it is defined as

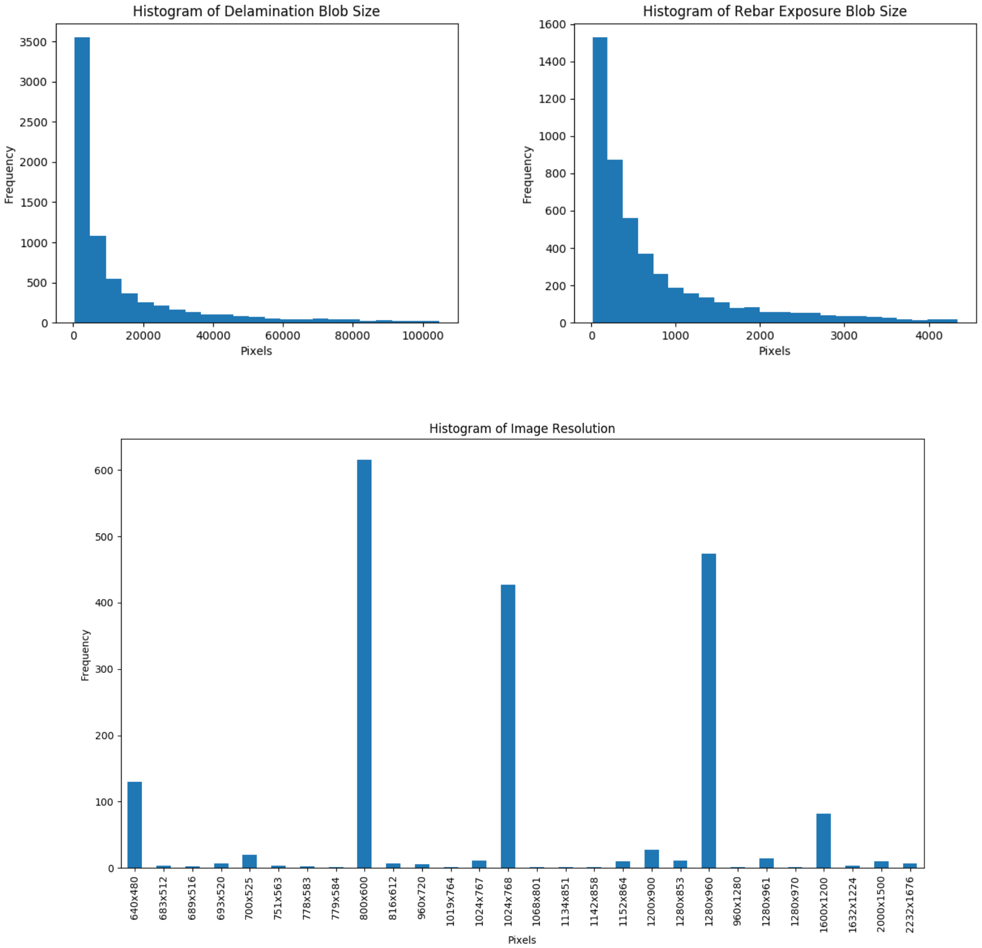


Figure 3: Histogram of blob pixel sizes. *Left* corresponds with rebar exposure and *right* corresponds to delamination blobs. *Bottom* shows a histogram of resolutions

the area inside the ROI which is not labeled as any of the two other damages. The average size of this class represents the 86% of the entire deck area. The images are in general around 1Mpx and most of them present a standard resolution while others are crops of larger images.

In order to evaluate the level of agreement among the labelers, in table 2 we have the per class IoU. This agreement has been computed by averaging the IoU of the 3 unique combinations of the labels per image across the entire dataset.

	Delamination	**Rebar exposure**	**Non-damage**
IoU	0.435	0.466	0.887

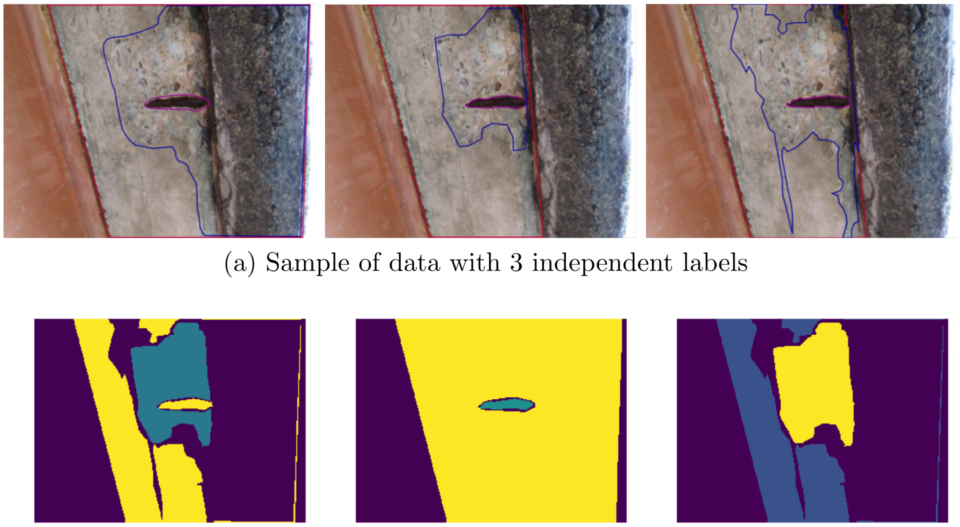
Table 2: Average IoU of the 3 labels per image reported per class.

We can observe how the agreement on non-damaged areas is very high, being generally easy to identify and in those parts in which there is no agreement tend to be small in relation to the entire area of non damage. On the other hand, delamination even though having an easy to identify texture, the lack of agreement is present when it comes to consider an area as delaminated, since depending on the degree of aggregation some inspectors could argue if it is or not a damage. This damage generally has a shared core among different labelers and the disagreement tends to be on the border of the damage. Finally, rebar exposure has a slightly higher agreement due to its very well defined elongated shapes but still lacks a high agreement due to the difficulty in defining a rebar exposure in some cases since depending on lighting and even dirt or concrete cracks, it can be misinterpreted as rebar exposure. In this case, the agreement tends to be more refined but the hit rate is lower than in the delamination blobs.

3.3 Multilabel Approach

Considering the nature of the dataset and its variability in interpretation of the damages, every image has a set of 3 labels obtained independently from different experts. This allows us to obtain a more accurate estimate of human level accuracy as well as a metric to consider areas of uncertainty. On the other hand, having multiple labels per image allows training the model with different approaches based on the degree of agreement among the different labels, from a strict way of only using total agreement regions to using the multiple labels as a data augmentation strategy yielding a dataset 3 times bigger by training every image sample with different labels.

In oder to take advantage of having multiple labels per data sample, the labels have been extended to form another ground truth (GT) in which the certain, uncertain and penalization regions for each class (C) are considered. This allows us to take into consideration the labeling agreement among the different labelers.



(a) Sample of data with 3 independent labels

(b) Extended label for delamination

(c) Extended label for rebar exposure

(d) Extended label for non damage

Figure 4: Multilabel example and corresponding extended label per class. Blue region corresponds to $GT_{C_{core}}$, yellow region corresponds to $GT_{C_{error}}$ and purple to $GT_{C_{uncertain}}$

$$GT_{C_{core}} = \cap_{i=1}^{L_C} GT_C \tag{1}$$

$$GT_{C_{uncertain}} = \cup_{i=1}^{L_C} GT_C \setminus GT_{C_{core}} \tag{2}$$

$$GT_{C_{error}} = x \not\in \cup_{i=1}^{L_C} GT_C \tag{3}$$

$GT_{C_{core}}$ is the main target of the prediction since it represents the total agreement. On the other hand, $GT_{C_{uncertain}}$ represents the area in which at least one label contains pixels of a particular class C but does not belong to the total agreement. Finally, $GT_{C_{error}}$ is the region in which there is total agreement of no presence of class C, and therefore, the region that will be penalized in the metrics for the predictions. An example of the resulting ground truth per class given a set of 3 different original labels can be seen in figure 4.

4 Deep Semantic Segmentation

Semantic segmentation is the task to assign semantic labels to regions of pixels in an image from a set of defined classes. This problem is an extensive topic in computer vision and has applications in a broad set of fields such as autonomous driving, medical imaging or robotics, among others.

In classic computer vision, most approaches rely on building features such as SIFT [22] or based on filter banks [17] to characterize pixels or regions and trained afterwards using machine learning supervised methods such as SVM or Random Forest.

In recent years, Convolutional Neural Networks (CNN) have been proven to outperform classic computer vision techniques in terms of accuracy and, in many cases, efficiency, for both classification and semantic segmentation tasks. Deep semantic segmentation is based on deep neural network architectures, mainly CNNs, and represent most of the state-of-the-art approaches for segmentation [11].

This section aims to describe the general concepts in a Convolutional Neural Network and review some architectures for semantic segmentation.

4.1 Convolutional Neural Networks

4.1.1 Introduction

Convolutional Neural Networks (CNNs) are a kind of artificial neural network targeted to grid-like topology structured data. This range from 1-D signals such as audio, 2-D signals such as images or 3-D signals encoding video or volumetric data. Even though it can be used in any kind of structured data, CNNs are mainly used in computer vision and gained a lot of attention in recent years, especially after the breakthrough of AlexNet [14] in the 2012 ILSVRC (ImageNet Large-Scale Visual Recognition Challenge).

Before describing deep semantic segmentation approaches we will describe in the following sections the main building blocks of a regular CNN.

4.1.2 Convolution

The convolution is the main building block in a CNN. A formal definition of a discrete convolution for a 2-D case can be found in equation 4. This can be easily generalized to N-D dimensions.

$$O(i,j) = (IK)(i,j) = \sum_{m}\sum_{n} I(i+m, j+n)K(m,n) \tag{4}$$

The output O or feature map is obtained via convolving an input I with a kernel K. The operation can be seen as sliding the kernel over the input volume and at each position, the output is the sum of product of the overlapping elements. The size of the output feature map is conditioned by the kernel size k, the stride s, the input size i and the padding p added to the input volume. We will consider for simplicity that the size of both axis are the same for the input feature map $s_i = s_j$ and the kernel $k_i = k_j$. The output is given by F filter or kernels of size o:

$$o = \left\lfloor \frac{i+2p-k}{s} \right\rfloor + 1 \tag{5}$$

An extensive description of different visual examples of different scenarios that influence the output size and the arithmetic behind, can be found in [7].

Convolutions hold three features that are inherent to the operation and can help improve a machine learning system: sparse interactions, parameter sharing and equivariant representations.

Unlike fully connected layers, in a convolution layer there is no total connectivity between output and input units. The values of an output feature map are given by only those inputs that interact with the filter in the convolution. This yields fewer parameters which decreases the amount of memory required as well as the number of operations. This feature is given by the fact of using filters of smaller size than the inputs.

Parameter sharing is a feature that also improves the memory requirements. The kernel is convolved with the input feature map and the same kernel values are used irregardless of the location. This behaviour is different than a fully connected network in which every output is connected to every input with an independent weight for every connection. Parameter sharing, thus causes the layer to have the property of equivariance to translation since the learnt filter is independent to the location.

Out of the main building blocks of a CNN and not considering the fully connected net appended for classification tasks, the convolution operations correspond to the learnable part of a convolution layer. The weights of the filters can be learnt through backpropagation.

4.1.3 Pooling

A regular convolution layer typically is a concatenation of operations: one or several convolutions, an activation function to introduce nonlinearities in the network and a pooling layer. The nonlinearity can be any kind used in fully connected networks, being a common choice in current architectures the rectified linear unit (ReLu).

The pooling layer maps the the feature map output from a convolution with a summary statistic of subregions defined by a given window size. The most used function is max pooling which summarizes each subregion with the maximum value. Other common options are average pooling or weighted pooling, among others.

This operation makes the network more robust in terms of small translation of the input. Another important use of pooling is the reduction in size of the feature maps, improving the memory usage and statistical efficiency. Pooling with downsampling is done by pooling regions with a stride s different than 1 pixel.

4.1.4 VGG Network

In the previous sections we described the main CNN building blocks, the convolution and pooling operations. In classification CNNs a set of convolution layers are concatenated in order to hyerarchically extract feature maps from the successive convolutions of the input, and finally a fully connected network is appended to perform classification on such features.

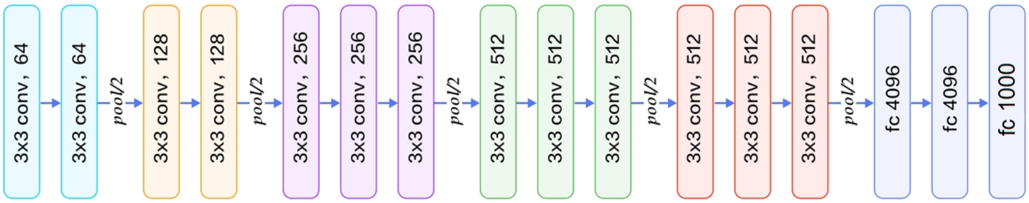


Figure 5: VGG-16 architecutre

VGG [33] is a network architecture developed by the Visual Geometry Group at Oxford University in 2014. One of the main characteristics of this particular architecture that popularized it is its simplicity. The use of smaller filter size of 3x3 compared to larger filters such as the 7x7 or 11x11 reduces the number of parameters and allowed to have an effective bigger size by stacking multiple convolution operations before max pooling. By stacking multiple convolution operations of size 3x3, without downsampling in between, the effective filter size is 5x5 for 2 stacked convolutions and 7x7 for 3 stacked convolutions, but with the advantage of fewer parameters. Another advantage of decomposing a convolution operation into multiple smaller-sized stacked convolutions is the ability to perform a non linearity after each of the convolutions, making the decision function more discriminative. The downsampling of the feature maps is performed via 2x2 maxpooling with a stride of 2 yielding a downsampling by factor of 2 at every maxpooling. In order to compensate the loss of spatial size of the feature maps after each downsample, the number of filters is increased by a factor of 2 after each downsample. Finally, all the activation functions used in all hidden layers are rectified linear units.

VGG-16 is one of the different architectures tested in [33] consisting of 16 weighted layers (13 convolutional + 3 fuly connected) and is one of the most pre-trained networks used due to its results and simplicity. The strategy used to reduce the parameters such as using stacked convolutions with small filter receptive fields allowed to stack multiple layers and reinforced the notion and importance of depth in terms of hyerarchical representations in deep learning.

4.1.5 Transfer Learning

Most of the deep semantic segmentation architectures are built partially making use of pre-existing architectures used in classification tasks. Thus, it is important to define the concept of transfer learning.

Training deep neural networks from scratch in many cases is an unfeasible task due to limitations in data samples. On the other hand, using pre-trained weights can speed up the convergence in comparison to training from randomly initialized weights [29]. Transfer learning has been proved [37] to work better as initialization even for dissimilar tasks.

Most networks used for transfer learning are trained on large datasets such as ImageNet [6]. Some examples of common networks used are AlexNet [14], VGG [33] or ResNet [12]. In closely related tasks, a common approach is to fine-tune partially the network, generally last layers, allowing to keep more generic features encoded in shallower ones.

4.2 Deep Semantic Segmentation review

4.2.1 Fully Convolutional architecture

One of the most popular methods for deep semantic segmentation are those architectures based on the concept of Fully Convolutional Network (FCN). An FCN is a neural network in which fully connected layers are convolutionalized in order to output dense predictions, which generally are upsampled via different techniques.

A more in depth description of the original work for Fully Convolutional Networks [21] based on a VGG-16 network is described in section 5.1.

4.2.2 Encoder-Decoder architecture

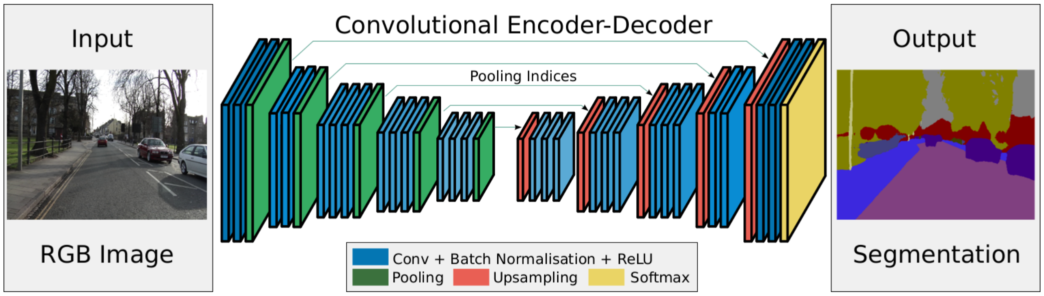


Figure 6: Representation of the ResNet architecture. Left part of the convergence point corresponds to the encoder based on VGG network. Right part corresponds to the decoder based on learnable convolutions and unpooling layers defined by its symmetrical pooling operation

Most networks for semantic segmentation make use of networks trained for classification tasks such as VGG or ResNet by discarding the fully connected layers and using the output feature maps from the convolutional layers. Thus, this part of the network is used to capture a subsampled representation of the input and is known as the encoder. The challenge then is to design a reliable mapping from the lower resolution encodings to a pixel-wise prediction.

SegNet [2] defines the encoding part of the network by using the convolution layers of VGG-16. By completely discarding the fully connected layers the number of trainable parameters in the encoder network is reduced by a factor of 10. Each of the convolution blocks are initialized with the weights of VGG-16 and define an encoder block, each of which is batch normalized and applied a ReLu non-linearity. Finally, the encoder block contains a max-pooling layer with a 2x2 window and stride of 2, yielding a subsampling by a factor of 2. While successive max-pooling operations improves the translation invariance, it comes at the cost of losing spatial resolution in the feature maps.

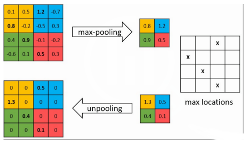
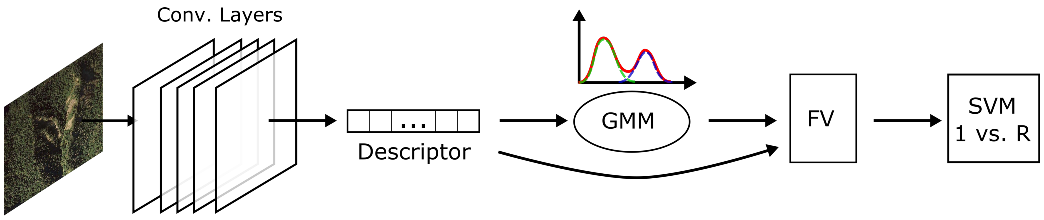


Figure 7

The decoder network is analogous to the encoder with unpooling or upsampling layers. This upsampling is performed by creating a sparse matrix making use of the corresponding decoder max-pooling indices. An example of this operation can be seen in figure 7. The indices of the max-pooled elements are kept from the max-pooling performed after the encoder and shared for the unpooling layer. This strategy allows a very efficient way to perform upsampling due to the low memory requirements for storing indices of max-pooled elements. After each unpooling layer, the sparse feature map is convolved with a trainable decoder and batch normalized. Finally, the last upsampled feature map is fed to a trainable soft-max, allowing a standard end-to-end training and pixel-wise prediction at inference.

4.2.3 Descriptors from feature maps



One way to make use of the encoding capabilities of a CNN is to pool feature maps from a pre-trained network in order to build descriptors. Generally, descriptor-based approaches are used in classification tasks but can be applied to segmentation via using a region proposal algorithm.

Deep Filter Banks [5] uses a VGG Network excluding the fully connected layers to pool the feature maps from the last convolutional layer. Then, these low level features are used to fit a Gaussian Mixture Model (GMM) from which a final Fisher Vector descriptor is built for each data sample by means of derivatives with respect to the GMM parameters. Finally, this highly dimensional descriptor is reduced with PCA and used to train a one-vs-all SVM.

This method is applied to texture recognition using the descriptors as well as for semantic segmentation by using region proposals from [1]. Object proposals from Multiscale Combinatorial Grouping (MCG) are assigned the label corresponding to the highest score from the SVM and they are pasted one by one in a ranked-fashion prioritizing high confidence regions and bigger areas.

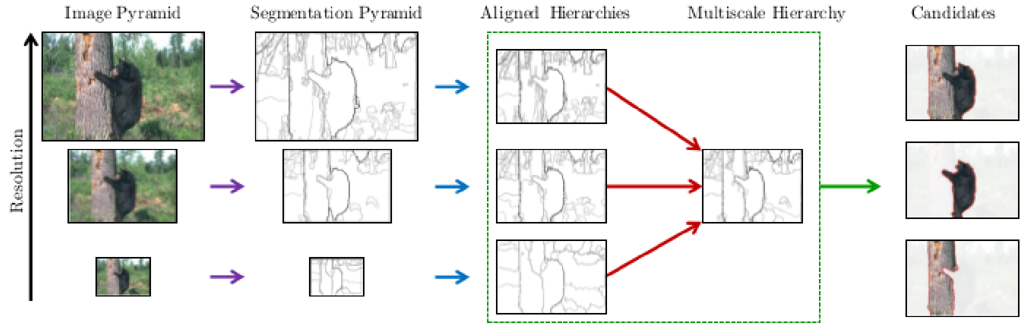


Figure 8: Example of region proposals from MCG [1]

One of the main drawbacks of this kind of approaches is computation time and the lack of end-to-end training and evaluation compared to other deep semantic segmentation methods.

5 Methodology

In this section we provide a deeper review of Fully Convolutional Networks, specifically the original work proposed in [21]. After a method description we proceed to explain our particular experiment setup on our dataset.

5.1 Fully Convolutional Network

Fully Convolutional Networks (FCNs) are one of the main approaches used in deep semantic segmentation. The main of this architecture is to use only convolutional layers to perform end-to-end training and inference of semantic segmentation 9

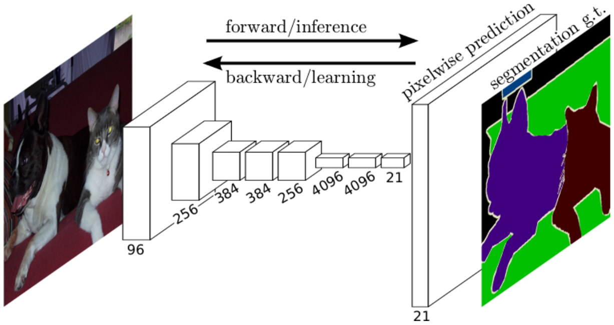


Figure 9: FCNs [21] provide an end-to-end framework for semantic segmentation

A regular neural network computes a nonlinear mapping function while a fully convolutional network can be seen as computing a nonlinear filter [21]. Due to its lack of fully connected layers and the nature of the convolution operation, an FCN works on any input size (only constraint to the number of downsampling operations).

There are three main concepts linked to FCNs: convolutionalization of dense networks, upsampling feature maps and adding shallower feature maps information via skip connections. These main concepts are explained in detail in the following subsections

5.1.1 Convolutionalization of Dense Networks

CNNs for classification lack spatial information in the outputs. The fully connected layer step loses track of spatial coordinates from the input and limits the input size of the network, forcing previous approaches to work on a patch-wise approach.

Fully connected layers can be viewed as convolutions with kernels that cover the entire input region, yielding one value per filter convolved. Adjusting the number of filters as units in the fully connected layer and following this 1x1 convolution covering the entire region of the input feature spaced, a fully connected layer can be convolutionalized.

Casting into convolutional layer a fully connected one results in output feature maps that are equivalent to the evaluation of the fully connected version on particular input patches. An example of such convolutionalization can be seen in figure 10

Quoting Yan LeCun on the matter of convolutionalization in [16]: “In Convolutional Nets, there is no such thing as ”fully-connected layers”. There are only convolution layers with 1x1 convolution kernels and a full connection table.”

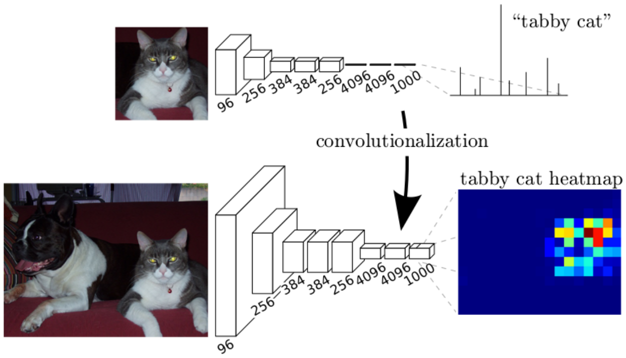


Figure 10: FCNs [21] provide an end-to-end framework for semantic segmentation

These heat map output makes this convolutionalization approach a natural move for semantic segmentation. Generally, in networks with pooling operations, the output feature maps represent a coarse version of segmentation due to the loss of resolution and are equivalent to evaluating the original net on particular input patches, while taking advantage of the optimization of convolution operations.

5.1.2 Upsampling Layer

The need of an upsampling layer comes from the loss of resolution in the successively downsampled feature maps in a fully convolutional network. As we explained in the previous section, by having convolution layers instead of fully connected layers the network outputs a coarse patch-equivalent segmentation.

Transposed convolution or, also called fractionally strided convolutions, work by swapping the forward and backward pass operation. Swapping this operation is the most common way to implement a transposed convolution for upscaling a feature map. As an example, a convolution of a filter of $3x3$ with a $4x4$ input volume with no padding and unitary stride results in a $2x2$. The transposed of such convolution would yield a $4x4$ from a $2x2$ input. This approach for upscaling allows to upsample in-network allowing end-to-end training through backpropagation.

The transposed convolutions can be initialized with interpolation methods such as bilinear interpolation, and can be kept fixed or learned as any other weight in the network.

5.1.3 Skip Connections

The coarse output heatmap from a convolutionalized regular CNN are generally upsampled to match the resolution of the input image. One issue with upsampling coarse feature maps is the loss of resolution in the final feature map due to consecutive downsamples, which yields in poorly detailed upsampled feature map.

Feature fusing allows to keep information from different depths of the network. Adding *skip connections* at different points of the network allow to combine feature maps before making the final upsample to restore the original image resolution. In figure 11 the configuration of FCN32, FCN16 and FCN8 is presented. In FCN32, after the convolutionalization of the fully connected layers, an upsampling layer providing x32 upsample to compensate the downsample of the pooling layers is applied. When adding a skip connection for FCN16 from *pool4*, the features are convolved with a prediction layer of 1x1 convolutions with output filters as the number of classes. This output is combined by means of summation with the x2 upscale of the feature maps of the convolutionalized last fully connected layer and finally the combination is upsampled by a factor of x16 resulting in the same input resolution. In a similar manner, FCN8 is built by fusing the feature maps of *pool3*

The models are trained sequentially, which means that an initial FCN32 is trained and used as weight initialization for training FCN16 with one skip connection and so on.

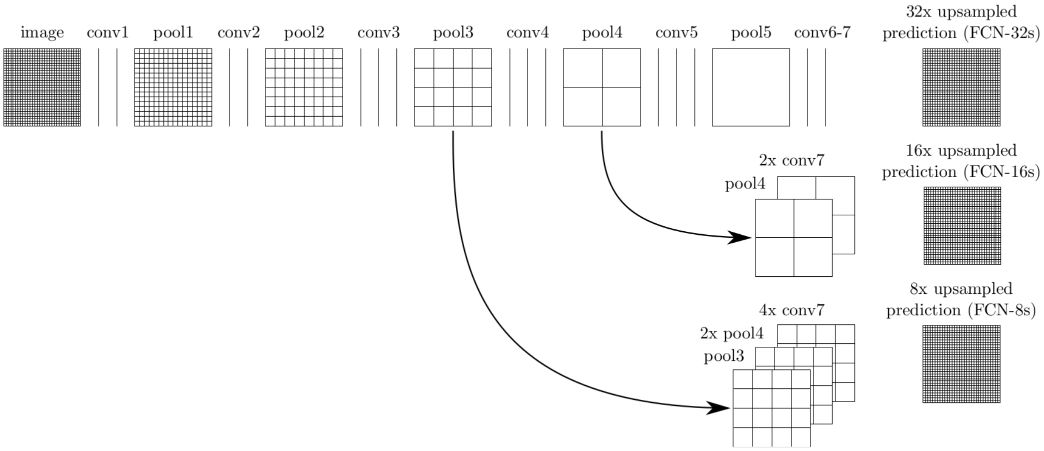


image
conv1
pool1
conv2
pool2
conv3
pool3
conv4
pool4
conv5
pool5
conv6-7
32x upsampled prediction (FCN-32s)
2x conv7
pool4
16x upsampled prediction (FCN-16s)
4x conv7
2x pool4
pool3
8x upsampled prediction (FCN-8s)

Figure 11: Skip connections on a VGG-16 network. Top row represents FCN32 (no skip connections). Middle row represents FCN16 (one skip connection from *pool4*) and bottom row FCN8 with skip connection from shallower *pool3*

5.2 Metrics and Evaluation

Jaccard distance or Intersection over Union (IoU) is a standard metric for assessing the performance of object segmentation models, both for bounding box and pixel-wise problems. This metric evaluates the level of agreement between the prediction and the ground truth by taking the ration between the intersection and the union of each mask. A formal definition for a particular class C is as follows:

$$IoU_C = \frac{TP_C}{TP_C + FN_C + FP_C}$$

This metric is computed by the ration of the number of overlapping pixels and the count of the union of the masks. This evaluation index yields a value between of 0 and 1 being the case where there is no overlap and exact overlap between the masks.

As explained in section 3 the use of multiple labels per image allows us to expand the labels to consider uncertainty in the ground truth. The evaluation metric used in the experiments consists in evaluating the IoU on the test set considering the regions described in 3 and, based on the original definition of IoU, the metric is defined in equation 6.

$$IoU_C = \frac{|GT_{C_{Core}}|}{|GT_{C_{Core}}| + |GT_{C_{Error}}|} \tag{6}$$

By using this metric we still have an analogous to IoU metric but not penalizing the region that is labeled as uncertain.

The results presented in 6 includes IoU per class C as well as the mean, both for the IoU taking into account the uncertainty regions as well as standard IoU for a fixed set of labels in order to compare performance.

5.3 Training Strategy

Two different training strategies have been tested regarding the multiple labels. A first and more strict approach is to train each image sample with the intersection among the labels, considering only the total agreement as the ground truth for an image. For the sample presented in figure 4 the corresponding label would be the label shown in 12

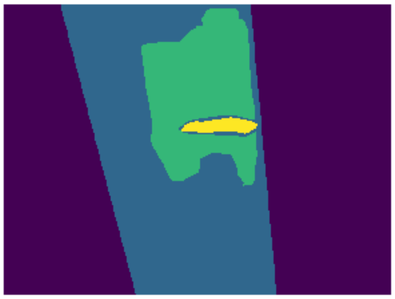


Figure 12: Label of sample of data based on the intersection of each original label. Green represents the intersection area of delamination, yellow the rebar exposure intersection, blue non damage intersection and purple represents the *don't care* area.

The second approach tested consists on using each of the multiple labels per image as a data sample. This can be seen as a way of data augmentation since each data sample in the training set is used with 3 different labels. In this case, during training, the label is randomly sampled from the pool of different labels.

5.4 Experiments

The dataset has been split in a 70/30 train and test ratio with 700 images in total (34 have been left out for future test set), randomly selecting instances of data while keeping the ratio of images from different levels of severity. For the purposes of this report this split has been performed due to time constraints. For the work-in-progress publication these results and potentially others will be crossvalidated. The results are preliminar results since more exploration in terms of experimentation is needed to conclude on best model.

The images are masked using the region of interest (ROI) of the deck area described in the dataset section. In order to keep consistency and to ignore small areas of no agreement on defining the region, the images and labels are masked taking the intersection of ROIs.

The model used is a Fully Convolutional Network based on VGG-16 for weight initialization.Trained with Stochastic Gradient Descent (SGD) on $256x256$ random crops with weight decay and momentum. The learning rate is doubled for biases and for every skip connection added the learning rate is decreased by a factor of 100. The training hyperparameters are defined as follows 3:

Parameters	**Value**
Batch size	8
Optimizer	SGD + weight decay
Learning rate	$1e-10$
weight decay	$1e-3$
Momentum	0.99

Table 3: Training parameters

Dataset augmentation is used by applying horizontal and vertical flips, change in illumination and slight affine transformations.

A total of 12 experiments have been tested by trying the training strategies stated in section 5.3, data augmentation and learnable or fixed upsampling transposed convolutions. These cases have been tested with up to 1 and 2 skip connections. Empirically we found that in all cases data augmentation helped significantly the model so for one and two skip connections (FCN16 and FCN8) only experiments with data augmentation have been trained. Following Long, Shelhamer, and Darrell the learnable upsamples are used in all the transposed convolutions except the last one which is a fixed bilinear upsampling. A summary of the experiments can be found in table 3

Experiment	**Model**	**Data Aug.**	**Learnable**	**Random**	**Intersec.**
1	FCN32			✓	
2	FCN32				✓
3	FCN32	✓		✓	
4	FCN32	✓			✓
5	FCN16	✓		✓	
6	FCN16	✓			✓
7	FCN16	✓	✓	✓	
8	FCN16	✓	✓		✓
9	FCN8	✓		✓	
10	FCN8	✓			✓
11	FCN8	✓	✓	✓	
12	FCN8	✓	✓		✓

‘’

Table 4: Summary of the models trained with the combinations of data augmentation, learnable upsample layers, and the two training strategies, random label or intersection of labels

The experiments have been run in a sequential manner, i.e. running models without skip connection first and using the weights as initialization to the next model with skip connection, and same procedure when adding an extra skip connection. The experiments with skip connections (FCN16 and FCN8) with a particular set of training parameters such as data augmentation or the training strategy have been initialized with the weights of its corresponding model trained on the same set of parameters.

The values reported correspond to the IoU considering uncertainty regions and the IoU over one out of three fixed set of labels.

6 Results

The first set of experiments correspond to the VGG-16-initialized Fully Convolutional Network in which the use of data augmentation and the two training strategies based on multiple labels are tested and no skip connections.

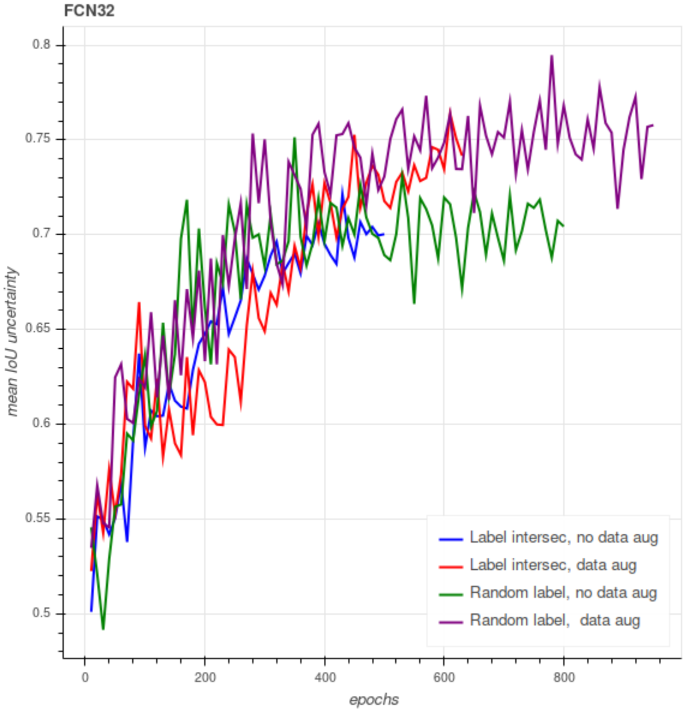


Figure 13: FCN32 models trained. The IoU explained in 5.2 is used for reporting the mean value of the 3 classes: non-damage, delamination and rebar exposure

From the initial set of experiments we can observe how data augmentation is effective in both training scenarios tested. The particular configuration that yielded better results for the case of no skip connections (FCN32) was obtained by training with data augmentation and randomly sampling from the pool of labels. As a reference, the validation standard mean IoU is computed on one of the sets in order to assess the difference in accuracy comparing to the extended IoU explained in section 3. We observe clearly how basing the evaluation on one label yields much lower results due to the difference in evaluation criteria from the civil engineer perspective.

Taking a closer look at per class values as we can see on table 5 we observe a marginal improvement on non damage class due to the high agreement among labelers. On the other hand, the highest increase is delamination over rebar exposure. By not having skip connections, the upsampling yields

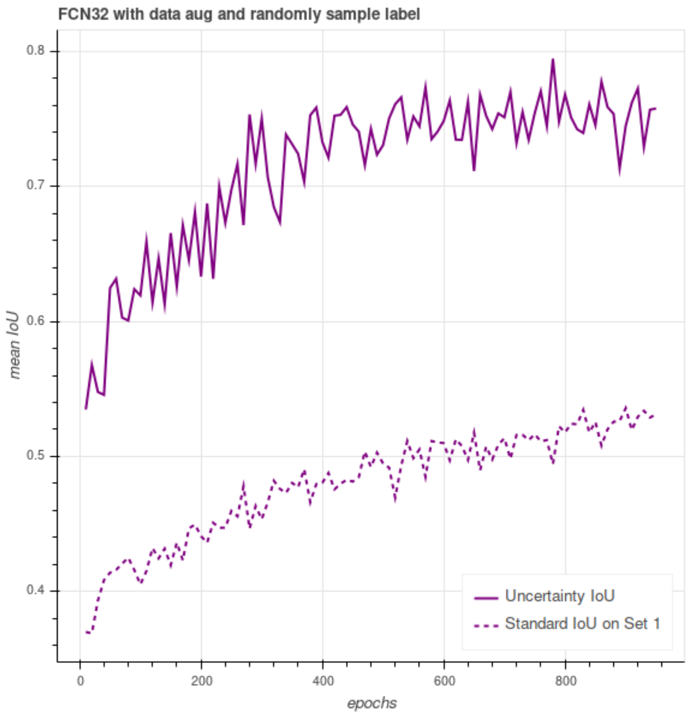


Figure 14: Experiment 4 from table 4. Comparison of mean IoU on a fixed set and mean IoU taking into account the certain and uncertain areas.

a coarse segmentation, and on delamination the tendency is to segment areas of bigger size than the ground truth. On the other hand, taking into account the uncertainty regions, this over-sized segmentation has a less negative effect due to the uncertainty region.

Clssses	**IoU set 1**	**Multilabel IoU**	**Delta IoU**
Non damage	0.89	0.97	0.08
Delamination	0.381	0.770	0.389
Rebar exposure	0.269	0.591	0.322

Table 5: Comparison of per class standard IoU on set 1 and IoU with multiple labels

Adding a skip connection, due to fusing lower level features in the output, initializing the network with the trained FCN with no skip connections and decreasing the learning rate has the effect of refining the segmentation. Due to data augmentation adding a positive effect on all previous experiments, data augmentation is applied to all FCN16 variants.

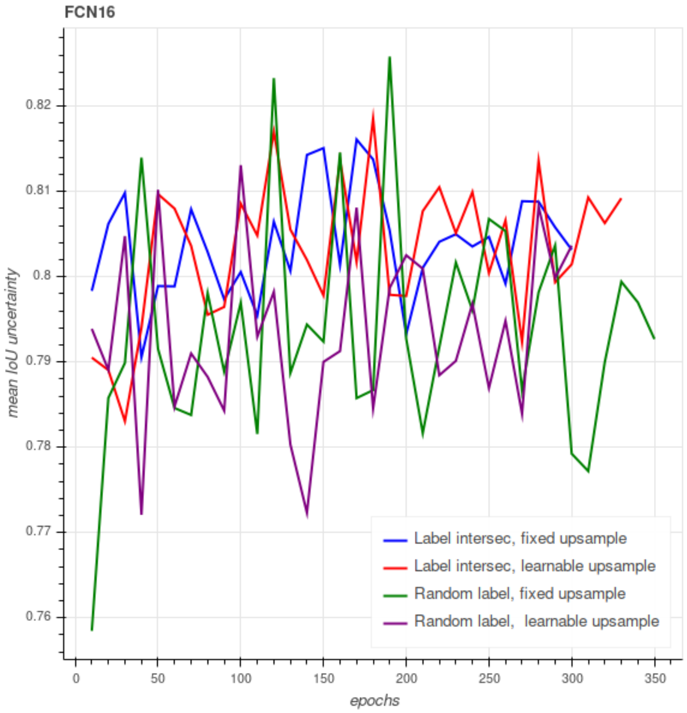


Figure 15: FCN16 models trained from the corresponding initialization from FCN32 models

Adding one skip connection makes all the models oscillate at 0.8 IoU. A common pattern is the marginal increase in IoU for learnable upscales even though in this case is almost a negligible improvement. On the other hand, for refinement with skip connection, training with the intersection of labels seems to be more effective, yielding slightly better results and less oscillation in the reuslts.

Similarly, we obtain the same behavior when adding a second skip connection (FCN8) with less overall improvement. The refinement of fusing multiple shallower feature maps tend to work on the refinement of the coarse segmentation and due to its slower learning rate, only marginal improvement can be obtained.

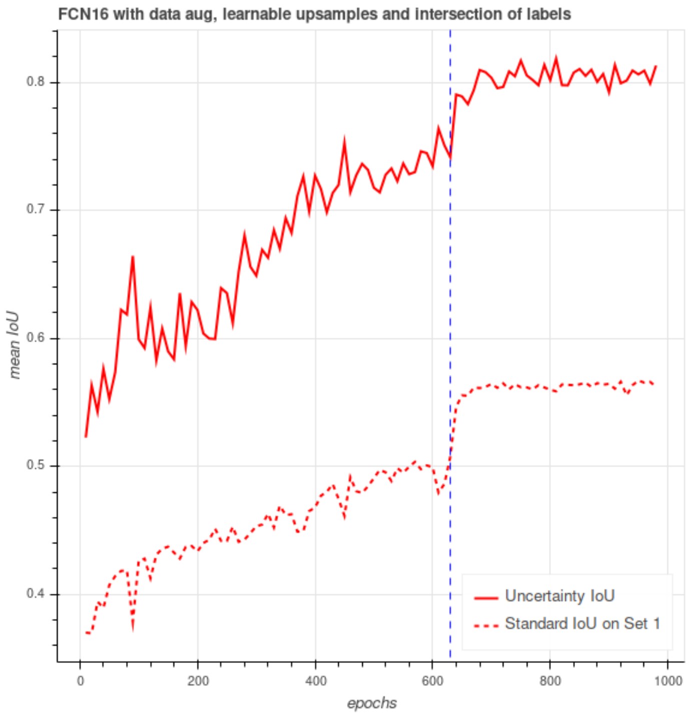


Figure 16: FCN16 with label intersection, data augmentation and learnable upsample. Mean IoU with uncertainty as well as standard mean IoU on fixed set 1 of labels is reported. Introduction of skip connection marked as vertical line in blue

In figure 18 the IoU of the different classes can be found. Non-damaged area presents no trouble in terms of segmentation and both the reference evaluation on one set and the uncertainty evaluation yields good results. Delamination can be detected from an early stage due to its larger on-average size compared to rebar exposure. On the other hand, rebar exposure due to its smaller size and tendency to get mistaken by areas with surface rust, takes more iterations to detect and benefits substantially from the addition of skip connections due to its well defined shape.

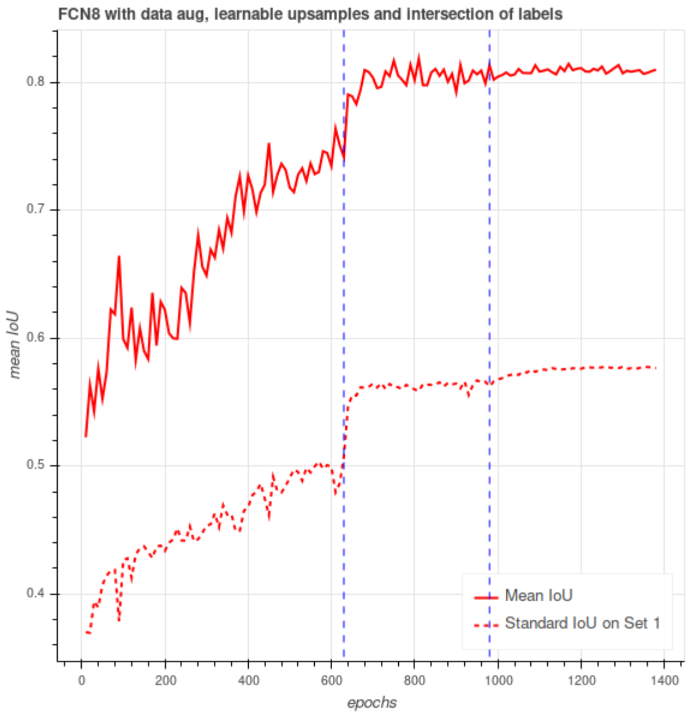


Figure 17: FCN8 with label intersection, data augmentation and learnable upsample. Mean IoU with uncertainty as well as standard mean IoU on fixed set 1 of labels is reported. Introduction of skip connection marked as vertical line in blue

7 Sample segmentation results

In this section we provide a few output inferred images from the 30% validation split used in the experiment section. The main idea is to illustrate the segmentation capabilities of such network on the images from bridge inspections, and to showcase some problem found empirically by inspecting the output results.

For the case of rebar exposure with delaminated area around the FCN shows very good performance. This kind of damage is the most severe case which mmkes it suitable for. aiding in the process of detection. An example of such case can be found in figure 19

The case of rebar with heavy delaminated area present a particular and less prone to uncertainty region. Thus, FCN is able to be very discriminative

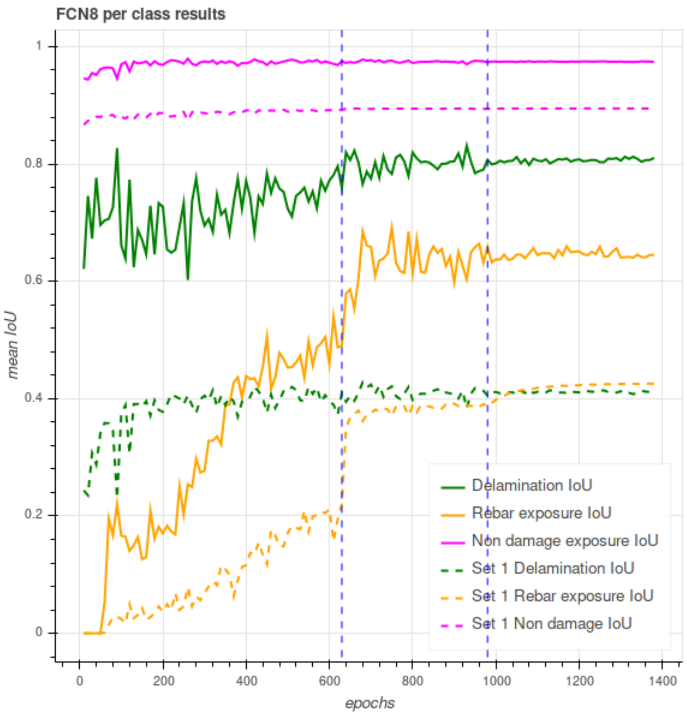
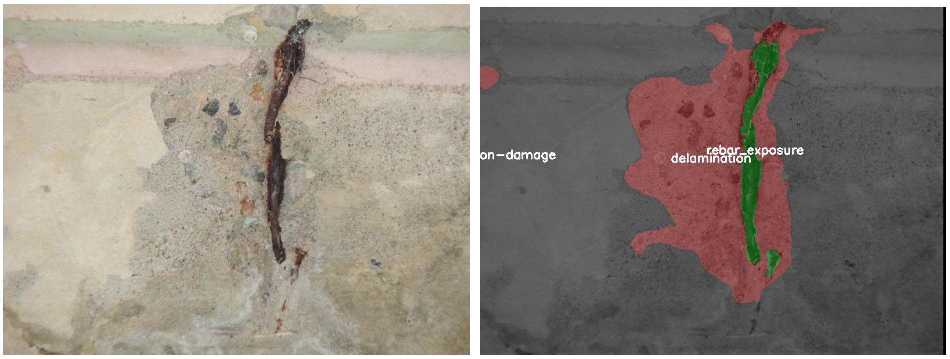


Figure 18: FCN8 results broken into classes. Both uncertainty based IoU and standard IoU over set 1 is presented

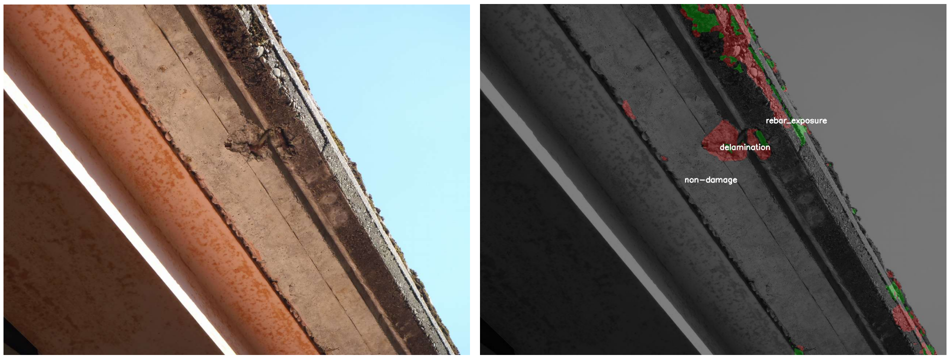


(a) Sample of data with 3 independent labels

Figure 19: *Left* original image. *Right* imge segmented with the FCN8 model stated in the experiments section

to this kind of pattern.

Regarding rebar exposure, it is the class more consistent with shape and texture. Generally it is a strong pattern easy to detect. On the other hand, one of the main problems for this particular class is the presence of false positives. From the texture perspective, any rust or certain types of dirt in the structure can easily be misclassified as rebar exposure. Further work to solve this problem has to be applied, being a simple postprocessing effective by removing small blobs of detected rebar exposure without surrounding delamination. A case of this particulr problem can be seen in figure 20



(a) Sample of data with 3 independent labels

Figure 20: *Left* original image. *Right* Image contining rebar false positives blobs segmented with the FCN8 model stated in the experiments section

8 Conclusions

Tackling a task in which the ground truth presents high variation requires from a way to quantify its uncertainty and perform a more accurate performance evaluation. In this work we contributed to the design of such metric based on regions of agreement from multiple labelers. Such dataset is a very valuable asset due to its multiple labels per image as well as for containing pixel-wise labeled images of bridges in the wild.

In particular, in relation to the structural damage segmentation we study the use of a VGG-based Fully Convolutional Network. This preliminary work shows promising results for semantic segmentation in the field of damage recognition and serves as a baseline for benchmarking other approaches once the dataset becomes public.

From the civil engineering perspective, at this stage, the model can be used as an aid for standard bridge deck inspection. Particularly in Japan, by law, the severity of damage in a bridge has to be assigned by a human which makes these kind of tools an aid to such task instead of an end-to-end tool. The work at this point serves as a proof of concept due to the limitation on classes (3 classes for this work) and it is being studied by [24] as a possibility to redefine the criteria to define the types of damage and the expansion of the dataset to include more types of damages as well as different parts of a bridge is being considered.

Thus, we conclude that FCNs and particularly VGG-based network pretrained on ImageNet it is suited for texture recognition in the scope of civil infrastructure surface damage semantic segmentation. While more effort has to be put into the data quantity and quality, as well as a deeper study of FCNs for this particular task, this work serves as a first step towards that goal.

References

[1] Pablo Arbeláez et al. “Multiscale combinatorial grouping”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition.* 2014, pp. 328–335.

[2] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. “Segnet: A deep convolutional encoder-decoder architecture for image segmentation”. In: *IEEE transactions on pattern analysis and machine intelligence* 39.12 (2017), pp. 2481–2495.

[3] Young-Jin Cha, Wooram Choi, and Oral Büyüköztürk. “Deep Learning-Based Crack Damage Detection Using Convolutional Neural Networks”. In: *Computer-Aided Civil and Infrastructure Engineering* 32.5 (2017), pp. 361–378.

[4] Jieh-Haur Chen et al. “A self organizing map optimization based image recognition and processing model for bridge crack inspection”. In: *Automation in Construction* 73 (2017), pp. 58–66.

[5] Mircea Cimpoi, Subhransu Maji, and Andrea Vedaldi. “Deep filter banks for texture recognition and segmentation”. In: *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on.* IEEE. 2015, pp. 3828–3836.

[6] Jia Deng et al. “Imagenet: A large-scale hierarchical image database”. In: *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on.* IEEE. 2009, pp. 248–255.

[7] Vincent Dumoulin and Francesco Visin. “A guide to convolution arithmetic for deep learning”. In: *arXiv preprint arXiv:1603.07285* (2016).

[8] A Ellenberg et al. “Bridge deck delamination identification from unmanned aerial vehicle infrared imagery”. In: *Automation in Construction* 72 (2016), pp. 155–165.

[9] Dongming Feng et al. “A vision-based sensor for noncontact structural displacement measurement”. In: *Sensors* 15.7 (2015), pp. 16557–16575.

[10] Y Fujino and DM Siringoringo. “Structural health monitoring of bridges in Japan: An overview of the current trend”. In: *Proc. 4th Int. Conf. FRP CICE.* 2008.

[11] Alberto Garcia-Garcia et al. "A review on deep learning techniques applied to semantic segmentation". In: *arXiv preprint arXiv:1704.06857* (2017).

[12] Kaiming He et al. "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[13] Klaudius Henke et al. "Use of digital image processing in the monitoring of deformations in building structures". In: *Journal of Civil Structural Health Monitoring* 5.2 (2015), pp. 141–152.

[14] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "Imagenet classification with deep convolutional neural networks". In: *Advances in neural information processing systems*. 2012, pp. 1097–1105.

[15] Rocco Langone et al. "Automated structural health monitoring based on adaptive kernel spectral clustering". In: *Mechanical Systems and Signal Processing* 90 (2017), pp. 64–78.

[16] Yann LeCun. *Yann LeCun on 1x1 convolutions*. `https://www.facebook.com/yann.lecun/posts/10152820758292143`. 2015.

[17] Thomas Leung and Jitendra Malik. "Representing and recognizing the visual appearance of materials using three-dimensional textons". In: *International journal of computer vision* 43.1 (2001), pp. 29–44.

[18] Gang Li et al. "Recognition and evaluation of bridge cracks with modified active contour model and greedy search-based support vector machine". In: *Automation in Construction* 78 (2017), pp. 51–61.

[19] Kuo-Wei Liao and Yi-Ting Lee. "Detection of rust defects on steel bridge coatings via digital image recognition". In: *Automation in Construction* 71 (2016), pp. 294–306.

[20] Yufei Liu et al. "Automated assessment of cracks on concrete surfaces using adaptive digital image processing". In: *Smart Structures and Systems* 14.4 (2014), pp. 719–741.

[21] Jonathan Long, Evan Shelhamer, and Trevor Darrell. "Fully convolutional networks for semantic segmentation". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 3431–3440.

[22] David G Lowe. “Distinctive image features from scale-invariant keypoints”. In: *International journal of computer vision* 60.2 (2004), pp. 91–110.

[23] *Ministry of Land Infrastructure Transport and Tourism.* `http://www.mlit.go.jp/en/index.html`.

[24] *Ministry of Land Infrastructure Transport and Tourism.* `https://www.nagai.iis.u-tokyo.ac.jp/index\_en.html`.

[25] YQ Ni, XW Ye, and JM Ko. “Monitoring-based fatigue reliability assessment of steel bridges: analytical model and application”. In: *Journal of Structural Engineering* 136.12 (2010), pp. 1563–1573.

[26] M. B. Nigro, S. N. Pakzad, and S. Dorvash. “Localized Structural Damage Detection: A Change Point Analysis”. In: *Computer-Aided Civil and Infrastructure Engineering* 29.6 (2014), pp. 416–432.

[27] Takafumi Nishikawa et al. “Concrete crack detection by multiple sequential image filtering”. In: *Computer-Aided Civil and Infrastructure Engineering* 27.1 (2012), pp. 29–47.

[28] Michael O’Byrne et al. “Regionally Enhanced Multiphase Segmentation Technique for Damaged Surfaces”. In: *Computer-Aided Civil and Infrastructure Engineering* 29.9 (2014), pp. 644–658. ISSN: 1467-8667.

[29] Maxime Oquab et al. “Learning and transferring mid-level image representations using convolutional neural networks”. In: *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*. IEEE. 2014, pp. 1717–1724.

[30] Roque A Osornio-Rios et al. “MUSIC-ANN Analysis for Locating Structural Damages in a Truss-Type Structure by Means of Vibrations”. In: *Computer-Aided Civil and Infrastructure Engineering* 27.9 (2012), pp. 687–698. ISSN: 1467-8667.

[31] Prateek Prasanna et al. “Automated crack detection on concrete bridges”. In: *IEEE Transactions on Automation Science and Engineering* 13.2 (2016), pp. 591–599.

[32] Bryan C Russell et al. “LabelMe: a database and web-based tool for image annotation”. In: *International journal of computer vision* 77.1-3 (2008), pp. 157–173.

[33] Karen Simonyan and Andrew Zisserman. “Very deep convolutional networks for large-scale image recognition”. In: *arXiv preprint arXiv:1409.1556* (2014).

[34] Xiao-Wei Ye, CZ Dong, and T Liu. “A review of machine vision-based structural health monitoring: methodologies and applications”. In: *Journal of Sensors* 2016 (2016).

[35] XW Ye, YH Su, and JP Han. “Structural health monitoring of civil infrastructure using optical fiber sensing technology: A comprehensive review”. In: *The Scientific World Journal* 2014 (2014).

[36] Chul Min Yeum and Shirley J. Dyke. “Vision-Based Automated Crack Detection for Bridge Inspection”. In: *Computer-Aided Civil and Infrastructure Engineering* 30.10 (2015), pp. 759–770. ISSN: 1467-8667.

[37] Jason Yosinski et al. “How transferable are features in deep neural networks?” In: *Advances in neural information processing systems*. 2014, pp. 3320–3328.

[38] Eduardo Zalama et al. “Road crack detection using visual features extracted by Gabor filters”. In: *Computer-Aided Civil and Infrastructure Engineering* 29.5 (2014), pp. 342–358.

[39] Allen Zhang et al. “Automated Pixel-Level Pavement Crack Detection on 3D Asphalt Surfaces Using a Deep-Learning Network”. In: *Computer-Aided Civil and Infrastructure Engineering* 32.10 (2017), pp. 805–819.