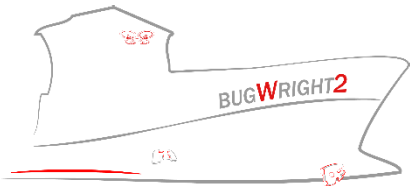
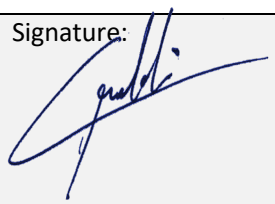


Autonomous Robotic Inspection and Maintenance on Ship Hulls and Storage Tanks

Deliverable report – D3.4

Context		
Deliverable title	Image Processing for Visual Inspection	
Lead beneficiary	UIB	
Author(s)	Alberto Ortiz	
Work Package	WP03	
Deliverable due date	31st March 2022 (M27)	
Document status		
Version No.	1	
Type	REPORT	
Dissemination level	**Public**	
Last modified	30 March 2022	
Status	RELEASED	
Date approved	30 March 2022	
Approved by Coordinator	Prof. Cédric Pradalier (CNRS)	Signature: 
Declaration	Any work or result described therein is genuinely a result of the BugWright2 project. Any other source will be properly referenced where and when relevant.	

This project has received funding from the European Union's Horizon 2020 research and innovation programme under Grant Agreement No. 871260

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

HISTORY OF CHANGES

Date	Written by	Description of change	Approver	Version No.
01/03/2022	UIB	Starting document		v0.0
24/03/2022	UIB	Report finished	CNRS	v1.0
29/03/2022	CNRS	Proofreading & validation	CNRS	V1.0

ABBREVIATIONS

CBC	*Coating Breakdown and Corrosion (detector)*
CNN	*Convolutional Neural Network*
CRF	*Conditional Random Field*
DCNN	*Deep CNN*
FCNN	*Fully Convolutional Neural Network*
FPSSD	*Feature Pyramid Single-Shot Multi-box Detector*
FSSS	*Fully-Supervised Semantic Segmentation*
IOU	*Intersection Over Union*
MCSS	*Multi-Class Semantic Segmentation*
MSE	*Mean Squared Error*
SSD	*Single-Shot MultiBox Detector*
WSSS	*Weakly-Supervised Semantic Segmentation*

Executive summary

This document describes novel deep-learning methods aiming at detecting defects in images in a visual way. The work described here focuses mostly on the detection of coating breakdown and corrosion (CBC). In this regard, we have developed two different methodologies: one that adopts an object detection approach based on bounding boxes regression while the other corresponds to a semantic segmentation algorithm, i.e. performs classification at the pixel level.

The contents of this report can be partly found in the following publications:

1. Kai YAO, Alberto ORTIZ, Francisco BONNIN-PASCUAL
   A DCNN-based Arbitrarily Oriented Object Detector for Quality Control and Inspection Applications, arXiv preprint arXiv:2101.07383v2, DOI: 10.48550/arXiv.2101.07383 (2021)
   (journal paper submitted, under review)
2. Kai YAO, Alberto ORTIZ, Francisco BONNIN-PASCUAL
   A Weakly-Supervised Semantic Segmentation Approach based on the Centroid Loss: Application to Quality Control and Inspection, IEEE Access, vol. 9, pp. 69010 – 69026 (2021)
3. Kai YAO
   Novel deep learning-based identification methods for accurate, orientation-aware visual detection with application to inspection and quality control
   PhD Dissertation, University of the Balearic Islands (2022)

I. Introduction

On the basis of the concept of Convolutional Neural Networks (CNN) proposed by LeCun and his collaborators (in the form of the well-known LeNet networks (Lecun, Bottou, Bengio, & Haffner, 1998)), followed by the technological breakthrough that allowed training artificial neural structures with a number of parameters amounting to millions (Krizhevsky, Sutskever, & Hinton, 2012), deep CNNs (DCNNs) have demonstrated remarkable capabilities for problems so complex as image classification, multi-instance multi-object detection or multi-class semantic segmentation. As it is well known in the research community, all this has been accomplished because of the "learning the representation" capability of CNNs. This capability is embedded in the set of multi-scale feature maps defined in their architecture through non-linear activation functions and a number of convolutional filters that are automatically learnt during the training process by means of iterative back-propagation of prediction errors between current and expected output.

In this report, we deal with the detection of one of the most common defects that can affect steel surfaces, i.e. coating breakdown and/or corrosion (CBC), in any of its many different forms. This is of particular relevance where the integrity of steel-based structures is critical, e.g. in large-tonnage vessels, storage tanks, etc. An early detection, through suitable maintenance programmes, prevents these structures from suffering major damage which can ultimately compromise their integrity and lead to accidents (with maybe catastrophic consequences, in the case of vessels, for the crew and passengers, environmental pollution or damage and/or total loss of the ship, its equipment and its cargo). The inspection of those structures by humans is a time-consuming, expensive and commonly hazardous activity, which, altogether, suggests the

introduction of defect detection tools to alleviate the total cost of an inspection. **Figure 1** shows images of metallic vessel surfaces affected by CBC.

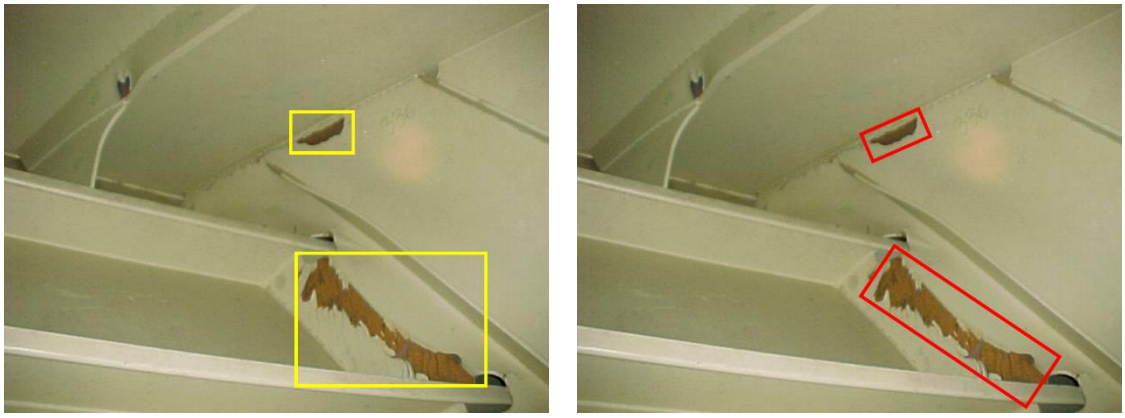
Figure 1: Examples of coating breakdown and corrosion (CBC) affecting ship surfaces.

In the following, we describe two different methodologies to address the visual inspection task and hence deal with the CBC detection problem: one that adopts an object detection approach based on bounding boxes regression (Section II) while the other corresponds to a semantic segmentation algorithm, i.e. performs classification at the pixel level (Section III).

II. Bounding-boxes regression-based approach

In this section, we adopt DCNN-based methodologies with an orientation towards multi-class object recognition, a domain for which DCNNs have shown very competitive performance under different operating conditions and with a minimum of human interaction or expert process knowledge. The proposal described in this work is a generic solution for multi-scale, arbitrarily-oriented object detection that can be applied to any context (after proper training). By *arbitrarily-oriented object detection* we mean that the output of the detector is a collection of oriented bounding boxes likely to contain any of the *objects of interest* for the task at hand. The fact that the detector is aware of objects orientation permits adapting the detection to the area where the object lies without involving more pixels from the background than necessary, thus producing a more effective detection (see **Figure 2** for an illustration). On the other side, a multi-scale detector allows dealing with objects that can appear in different sizes. Both features, multi-scale and oriented-detection, become necessary when the detection problem involves objects with irregular shapes and different sizes and aspect ratios.

In more detail, this section describes a two-stage arbitrarily-oriented detector: the first stage predicts locations for the objects of interest in the form of un-oriented bounding boxes, adopting a feature pyramid-based approach to produce detections at different scales and so capture minor details if needed. The second stage implements a lightweight CNN that is used for regressing the parameters of the oriented bounding boxes better fitting the objects of interest that lie inside the un-oriented predictions produced by the first stage.



(a) (b)

Figure 2: Use of oriented bounding boxes for objects with different shapes, sizes and aspect ratios: (a) example of detection by means of unoriented bounding boxes, (b) example of more effective detections by means of oriented bounding boxes.

The main contributions of this work are as follows:

- We design a two-stage arbitrarily-oriented multi-category object detector, which we show can successfully operate in the intended scenarios;
- We propose a feature pyramid-based network architecture and analyse several map fusion strategies;
- The unoriented boxes regressor adopts a default boxes-based scheme using the output of a process clustering the training data to obtain high-quality priors and improve target localisation accuracy;
- Oriented bounding boxes regression is achieved by means of a simple network;
- The evaluation performed includes comparative studies on some important design choices.

The rest of this section is organised as follows: Section 2.1 overviews the full network, while Section 2.2 describes the multi-scale, orientation-unaware detector, Section 2.3 outlines the default boxes selection process and Section 2.4 details the network producing oriented bounding boxes; finally, Section 2.5 reports on the results of a number of experiments aiming at showing the performance of the full detector.

2.1. Detector overview

The detector proposed in this work comprises two stages. The first stage is intended to regress unoriented bounding boxes by means of a variant of the *Single-Shot MultiBox Detector* (SSD). SSD is a one-stage object detection approach that makes use of the standard VGG-16 network as backbone though modified by replacing the last fully connected layers with the incorporation of additional convolutional layers (see (Liu, et al., 2016) for the details). In comparison with most detection algorithms based on R-CNN, such as (Girshick, 2015) and (Ren, He, Girshick, & Sun, 2015), SSD does not require any extra procedure to generate

proposals. Alternatively, a mechanism of prior boxes is used, from which offsets are regressed for enhanced localisation accuracy. On the other side, unlike R-CNN methods, detections are obtained at several scales from a number of layers of the backbone, namely *conv4_3*, *fc7*, *conv8_2*, *conv9_2*, *conv10_2* and *conv11_2*. The corresponding feature maps are subsequently involved in the calculation of a multi-term loss function to regress the parameters of the bounding boxes (i.e. offsets relative to the prior boxes shape), and to obtain confidence values for the classes. To this end, predicted boxes have to be matched with true bounding boxes to train the detector, and only those positives with enough overlap contribute to the loss, while positives and negatives contribute to the classification loss (after a hard negative mining process to keep the positive *vs* negative samples ratio at 1:3).

For the first stage of the detector, in this work, we follow the proposals-free approach of SSD together with the selection of a set of prior boxes, though with a number of differences: (a) the backbone consists in a pyramid of feature maps involving information at more scales than SSD, as depicted in **Figure 5**, to favour the detection of both large and small targets; (b) the pyramid involves a map fusion scheme that leads to the best performance among a total of four alternatives; and (c) the set of prior boxes are not arbitrarily hand-picked but the selection is guided by the training data, resulting from a clustering procedure taking the ground truth as input. The details can be found in Sections 2.2 and 2.3.

The second stage of the detector consists in a specifically designed network trained to regress the parameters of the rotated bounding box maximally contained in the unoriented bounding boxes stemming from the first stage. A detailed description of this stage is given in Section 2.4.

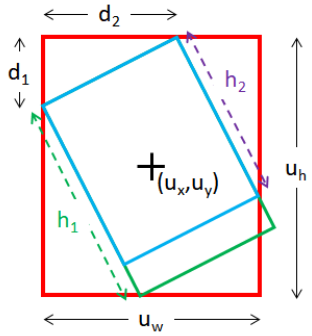


Figure 3: Parameterisation of oriented and unoriented bounding boxes: a $(\boldsymbol{d_1}, \boldsymbol{d_2})$ pair can lead to two different oriented boxes, with heights $\boldsymbol{h_1}$ and $\boldsymbol{h_2}$.

To finish, **Figure 3** illustrates how bounding boxes are parametrised in the approach proposed in this section. For unoriented bounding boxes, we make use of the standard parameters, namely the box centre (u_x, u_y) and its width u_w and height u_h. Regarding oriented bounding boxes, they are expressed in terms of the unoriented bounding box they are defined in, by means of intercepts (d_1, d_2). As shown in **Figure 3**, these intercepts result from the intersection between the rotated box sides and the unrotated box sides. Since this parametrization can lead to two different rotated boxes, a third optional parameter h can be included in the definition of the oriented bounding box to disambiguate between h_1 and h_2.

2.2. Feature Pyramid Single-Shot Multi-box Detector (FPSSD)

SSD uses feature maps from different layers of the network to regress bounding boxes. More precisely, SSD adopts large-scale feature maps to detect small targets, and conversely uses small-scale feature maps to detect large targets. In this work, we additionally make use of the *feature pyramid* concept to fuse feature

maps from top layers with feature maps from bottom layers to obtain enhanced features containing both semantic information and detailed features, which is exploited to detect different scale targets.

The idea of the feature pyramid originates from the *image pyramid* concept, which aims at being able to analyse an image at multiple scales by means of multi-scale sampling of the original image via e.g. Gaussian kernels. As assisted by a hierarchical CNN, a *feature pyramid* can be built in one single feed-forward pass that *simultaneously* calculates the multi-scale features of the input image. Hence, the feature pyramid can efficiently address the multi-scale problem with a relatively low cost.

So far, several works have implemented the feature pyramid concept onto DCNNs (see (Li & Zhou, 2017) and (Fu, Liu, Ranga, Tyagi, & Berg, 2017), among others). The four typical approaches for fusing the feature maps are overviewed in **Figure 4**. **Figure 4**(a) illustrates the most common strategy, FPN, which merges feature maps layer by layer by element-wise addition and performs detection from each scale/feature map. Another method is the lightweight fusion strategy named FSSD shown in **Figure 4**(b). In this case, features from different layers at different scales are concatenated together first and used next to generate a series of pyramid features. Lastly, the different feature maps are combined by the concatenation layer and sent to the loss function. Though this method is capable of saving computational costs as compared to method (a), the feature maps feeding the detector finally lack certain semantic information. **Figure 4**(c) illustrates FPSSD, the method proposed in this work, which employs a strategy identical to FPN to fuse the feature maps, but aims at reducing the computational cost by means of a concatenation layer that combines the different feature maps. Subsequently, the combined feature maps are fed into the detector. Lastly, **Figure 4**(d) depicts the strategy adopted in the original SSD. Among others, it shows that SSD does not integrate any feature fusion module, and thus it has a limited capability to capture simultaneously low-level details and high-level semantic data.

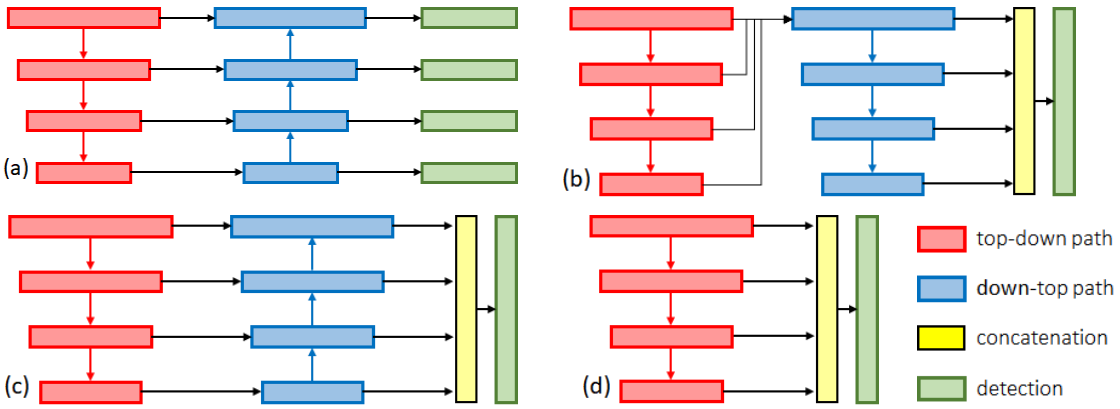


Figure 4: Different strategies for fusing feature maps in a feature pyramid:
(a) feature maps are fussed from top to bottom layer by layer; (b) a lightweight architecture that merges feature maps from top to bottom; (c) FPSSD;
(d) original SSD approach, which uses feature maps from different layers separately.

Figure 5(top) outlines the architecture of FPSSD. As can be observed, the feature maps are extracted from the *conv4_3*, *fc7*, *conv6_2*, *conv7_2*, *conv8_2*, and *conv9_2* layers of the original SSD network (Liu, et al., 2016). On the other side, deconvolution layers are utilised to enlarge the respective feature maps. We also make use of 1×1 convolutional layers, termed as lateral connections in (Lin, et al., 2017), to unify the output channels of all feature maps. Lastly, down-top layers integrate different scales and submit the result to the detector to predict the category and localisation of targets.

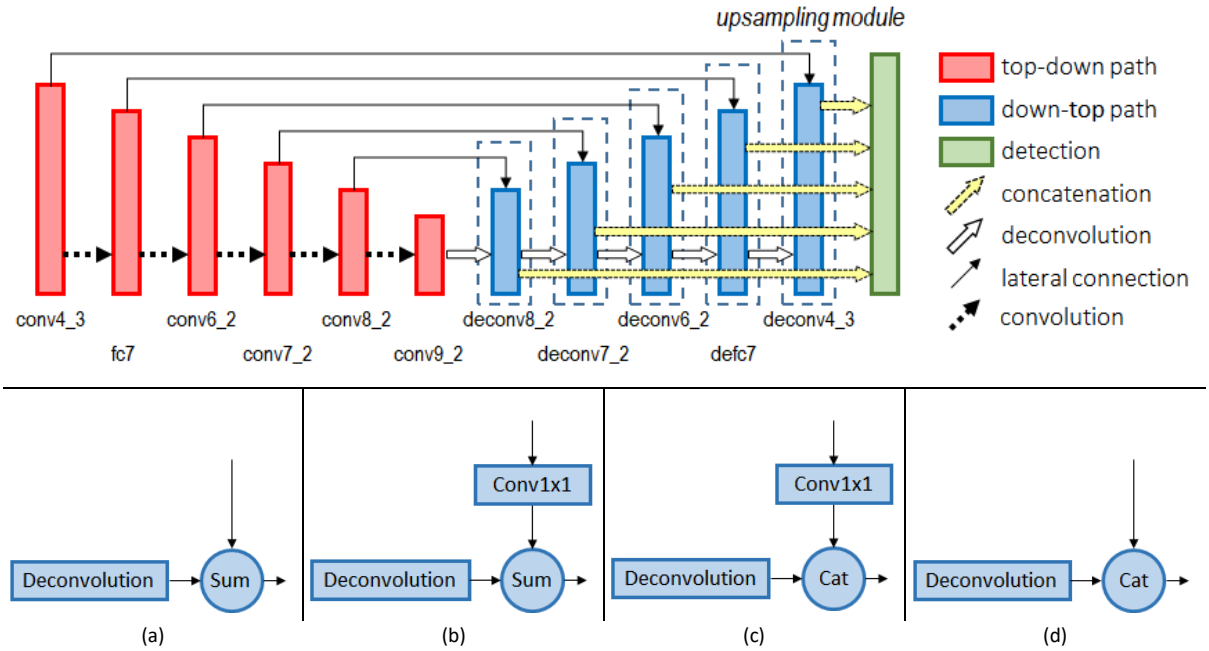


Figure 5: FPSSD: (top) architecture, (bottom) alternative implementations of the upsampling modules (Sum, Cat and Conv1×1 respectively denote pixel-wise sum, concatenation and 1×1 convolution).

2.3. Selection of (unoriented) Prior Boxes

SSD predefines a total of 6 prior boxes per feature map location by imposing different size combinations (w_k, h_k) manually picked. Since, on the one hand, the shape of the bounding boxes to detect can vary significantly and, on the other hand, SSD regresses the predicted bounding boxes from the prior boxes, a proper selection of those prior boxes becomes crucial for achieving a high detection success; as already noted in (Redmon, Divvala, Girshick, & Farhadi, 2016), such a proper selection contributes to the stability of the underlying optimization process, converges faster and improves effectively the *Intersection over Union* (IOU) between predicted and true boxes. Hence, our object detector makes use of prior boxes selected automatically in accordance to the available data.

In more detail, we run the well-known K-means algorithm over the bounding boxes belonging to the ground truth, using box width and height as the clustering features. Instead of the Euclidean distance, typically used by K-means implementations, we define IOU as a distance metric because we have observed better clustering results with the latter. The distance between a sample box b_i and the cluster centroid c_j is hence defined as:

$$d(b_i, c_j) = 1 - \text{IOU}(b_i, c_j) = 1 - \frac{b_i \cap c_j}{b_i \cup c_j} = 1 - \frac{o(b_i, c_j)}{a(b_i) + a(c_j) - o(b_i, c_j)} \quad (1)$$

where $o(\cdot,\cdot)$ denotes overlapping area and $a(\cdot)$ denotes area.

Table 1 shows averages of the IOU metric (see Section 2.5.2) for hand-picked prior boxes and automatically selected boxes by clustering, and different amounts of prior boxes (for the hand-picked cases, we predefine the boxes similarly to SSD). We can see that 4 clusters automatically selected yield similar performance than 10 hand-picked prior boxes. This means that it is possible to propose automatically higher-quality and better parameterized prior boxes. As could be expected, the more clusters, the better is the performance

(the trend can be observed to continue for 7 or more prior boxes), although the number of clusters should not be high to keep reasonable the running time.

Table 1: Mean IOU (mIOU) vs number of prior boxes and selection method.

Approach	no. prior boxes	mIOU (%)
Hand-Picked	4	35.93
Hand-Picked	5	37.96
Hand-Picked	6	42.75
Hand-Picked	10	61.82
Clustering	4	61.58
Clustering	5	63.37
Clustering	6	**65.31**

2.4. Regression of Oriented Bounding Boxes

To regress the parameters of the rotated boxes, a lightweight convolutional network based on LeNet has been adopted. With regard to the original network, the rotated boxes (RBox) regression network exhibits several differences: (1) the input size is 63×63 after the incorporation of an additional convolutional layer at the beginning of the network, in order to avoid reducing the image to LeNet's 28×28 input pixels and lose information; (2) batch normalization is used after each convolutional layer to speed up convergence during training (this has also been shown to decrease the effect of covariate shift from the hidden layers (Ioffe & Szegedy, 2015)); (3) since the bounding boxes parameters (d_1, d_2, h) range from 0 to 1, a sigmoid layer lies between the last fully connected layer and the loss layer; and (4) lastly, an Euclidean distance loss layer is used during regression:

$$L(d,g) = \frac{1}{2N}\sum_{i=1}^{N}\left(d_1^i - g_{d_1}^i\right)^2 + \left(d_2^i - g_{d_2}^i\right)^2 + \left(h^i - g_h^i\right)^2 \quad (2)$$

where $d = (d_1, d_2, h)$ denotes the predicted offsets and height, $g = \left(g_{d_1}, g_{d_2}, g_h\right)$ represents the ground truth and N is the size of the mini-batch. The architecture of the RBox regression network can be found in **Figure 6**.

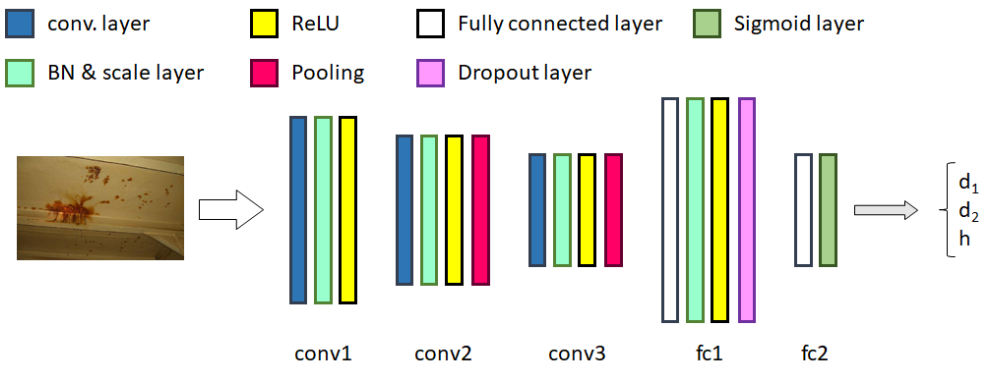


Figure 6: Architecture of the RBox regression network.

2.5. Experimental Results and Discussion

2.5.1. Experimental setup

The FPSSD and RBox networks have been implemented using Caffe (Jia, et al., 2014). Referring particularly to FPSSD, the VGG-16 network is taken as the backbone, identically to the original SSD. Trainings and all experiments have been performed on a PC platform fitted with an Intel i9-9900K processor with 64Gb RAM and an Nvidia RTX 2080Ti GPU.

The dataset employed for training comprises images from different vessels taken under different illumination conditions, viewpoints, etc. All the images have been resized to 512×512 pixels. As for training, we have adopted a multiple steps strategy, where the learning rate was set to 10^{-5} during the first 8000 iterations, the next 6000 iterations used a learning rate of 10^{-6}, and the final 6000 iterations employed a learning rate of 10^{-7}. The batch size was set to 10, which is the best configuration for the GPU involved in the experiments. We have employed SGD for network optimization, and the weight decay and the momentum were set to 0.001 and 0.9, respectively. As a compromise between accuracy and computation time, object detection was performed using six prior boxes whose features resulted from the clustering process described in Section 2.3.

2.5.2. Assessment metrics

We employ the following metrics for performance evaluation:

- For both unoriented and oriented bounding boxes, we consider the Recall (R), the Precision (P) and the Average Precision (AP) measured as the area under the P-R curve for a set of pre-defined recall values (Everingham, et al., 2015). Detected bounding boxes with a confidence above 0.7 have been considered as the set of predictions P of the detector (as usual for object detection).
- For the unoriented bounding boxes, we also consider the averaged IOU (AIOU):

$$\text{AIOU} = \frac{1}{|P|} \sum_{b_j \in P} \text{IOU}\left(b_j, g_j\right) = \frac{1}{|P|} \sum_{b_j \in P} \frac{b_j \cap g_j}{b_j \cup g_j} \tag{3}$$

  where $|P|$ stands for the cardinality of set P, b_j denotes a prediction and g_j is the true bounding box with highest overlap with b_j.
- To determine the performance of oriented detection, we also provide the averaged RBox IOU (ARIOU) as supplementary performance metrics (analogously to Equation (3)). Unlike the case of unoriented bounding boxes, the shape of the intersection of two rotated bounding boxes turns out to be into a convex polygon. In general, the area A_{cp} of such a polygon is given by:

$$A_{\text{cp}} = \frac{1}{2} \sum_{i=1}^{n} (x_i y_{i+1} - x_{i+1} y_i) \quad \text{[Shoelace Formula]}$$

  where $\{(x_1, y_1), \ldots, (x_n, y_n)\}\}$ are the coordinates of the polygon vertices arranged counter-clockwise, and $(x_{n+1}, y_{n+1}) = (x_1, y_1)$.
- To finish, in order to measure the accuracy of the parameters regressed, we also adopt the mean absolute error (MAE) for the regression targets considered, calculated as follows:

$$\text{MAE}_t = \frac{1}{|P|} \sum_{P} \left| t_p - t_g \right|$$

  where t_p and t_g respectively denote the predicted target value and the ground truth.

2.5.3. Regression results for unoriented bounding boxes

In this section, we report on the performance obtained for unoriented bounding boxes detection. We start with an ablation study considering the effect of the lateral connections between layers of the top-down and down-top paths and the necessary map fusion approaches. We consider SSD 512 as a baseline and the alternatives that are enumerated in **Figure 5** (bottom), which contemplate pixel-wise sum and concatenation for map fusion, and the use or not of 1×1 convolutional filters to unify the number of output channels from top to bottom layers. Results for different metrics are reported in **Table 2**. As can be observed, option (b) attains the largest performance in all cases.

Table 2: Ablation study: effect of lateral connections and the feature map fusion approach. (Bold face denotes best.)

Configuration	Figure 5(bottom)	mRec	mPrec	F_1	mAP
SSD 512		0.8311	0.9434	0.8837	0.8218
FPSSD 512 + Sum	(a)	0.8241	0.9513	0.8831	0.8131
FPSSD 512 + 1×1 conv + Sum	(b)	**0.9113**	**1.0000**	**0.9536**	**0.9091**
FPSSD 512 + 1×1 conv + Cat	(c)	0.8264	0.9433	0.8810	0.8133
FPSSD 512 + Cat	(d)	0.8262	0.9563	0.8865	0.8172

To finish, some qualitative results from a selection of images can be found in **Figure 7**. As can be noticed, FPSSD achieves competitive results, being particularly able to detect small areas affected by corrosion.

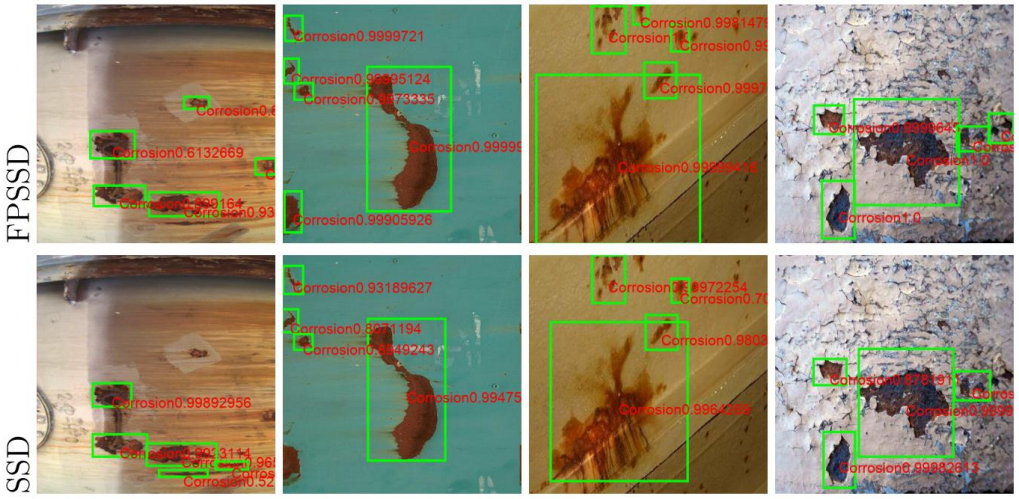


Figure 7: Unoriented detection results for FPSSD and SSD.

2.5.4. Regression results for oriented bounding boxes

Though FPSSD leads to good performance for unoriented detection for the visual inspection task, for some elongated targets, either regularly-shaped or irregularly-shaped, the results of FPSSD can be inaccurate, apart from the fact that unoriented bounding boxes tend to include parts of other objects and even cover a large fraction of the image in order to fully contain certain objects of interest, as shown in **Figure 7**. This are the reasons why oriented bounding boxes are considered in this work. In this section, we analyse the performance of the RBox regression network described in Section 2.4.

Table 3 shows the MAE for each regression target and two configurations: (a) two-target regression (d_1, d_2) and (b) three-target regresssion (d_1, d_2, h). As can be observed, the MAE values for d_1 and d_2 for the two-target case are lower than the corresponding MAE values for the three-target case. Moreover, the average MAE of the two-target case is also lower than the average MAE for the three-target case. On the

other side, we have fine-tuned the last fully-connected layer of AlexNet (Krizhevsky, Sutskever, & Hinton, 2012), i.e. we have only optimised the weights of the last fully-connected layer, the weights of the other layers have been frozen. We have also replaced the softmax layer by a sigmoid layer and used the resulting model as a baseline to compare with. **Table 3** shows that, on average, the fine-tuned AlexNet produces worse predictions than the RBox network.

Table 3: MAE values for the regression targets considered by the RBox network in comparison with AlexNet. (Bold face denotes best.)

Approach	d_1	d_2	h	average
RBox (2-target)	**0.1556**	**0.1612**	-	**0.1584**
RBox (3-target)	0.3151	0.3105	**0.0889**	0.2381
AlexNet (2-target)	0.1722	0.1915	-	0.1818
AlexNet (3-target)	0.2722	0.3744	0.2501	0.2989

Figure 8 shows some examples of detections of rotated bounding boxes for two- and three-target regression. In the pictures, the red points correspond to the d_1 and d_2 intercepts, while the green line represents the third regression target h. The black line just connects the red points to show the predicted orientation of the object detected. It can be observed that the black lines in the first column (using two-target regression) adhere better to the orientation of the objects than the detections of the second column (using three-target regression); the two-target RBox network outperforms as well the fine-tuned version of AlexNet.

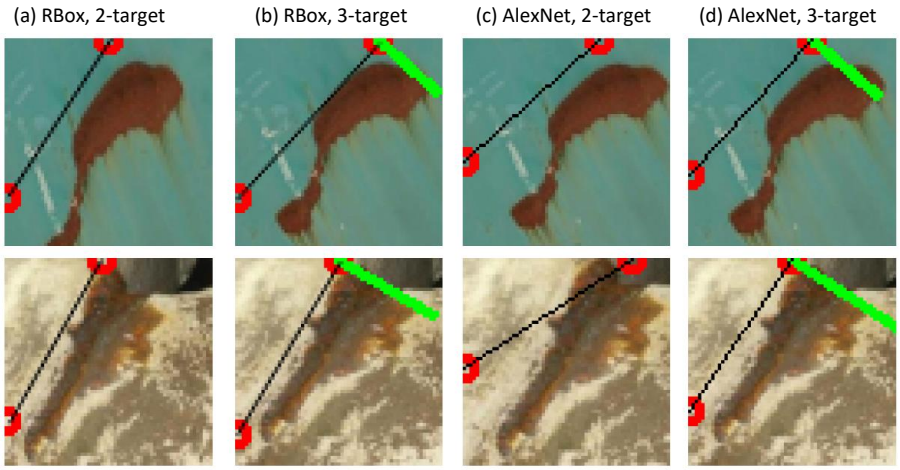


Figure 8: RBox regression results: (a) RBox network for 2 regression targets, (b) RBox network for 3 regression targets, (c) AlexNet for 2 regression targets, (d) AlexNet for 3 regression targets. (The red dots correspond to regression targets d_1 and d_2, while the green line represents the regression target h.)

At last, we connect the FPSSD and the RBox networks to infer oriented detections end to end, i.e. the input of the RBox regression network is the prediction of FPSSD. In this regard, notice that, because the output of FPSSD is a prediction, it could be slightly displaced with regard to the true object location (which is what has been used for training), increasing hence the challenge of estimating correctly the object orientation.

Table 4 compares, by means of ARIOU values, the two-target and three-target RBox regression networks with TextBoxes++ (Liao, Shi, & Bai, 2018), an arbitrarily-oriented text detector also based on SSD, which we have fine-tuned for the visual inspection task and employed as a baseline for this experiment. TextBoxes++ is combined with a neural network-based text recognition module which, for obvious reasons, is not involved in this experiment. On the other side, as already mentioned, the two-target regression variant of

RBox gives rise to two predictions (as described in **Figure 3**). For this case, we always select the largest oriented box.

Table 4: ARIOU values for the RBox network. (Bold face denotes best.)

RBox (2-target)	RBox (3-target)	TextBoxes++
0.5932	0.5419	0.4615

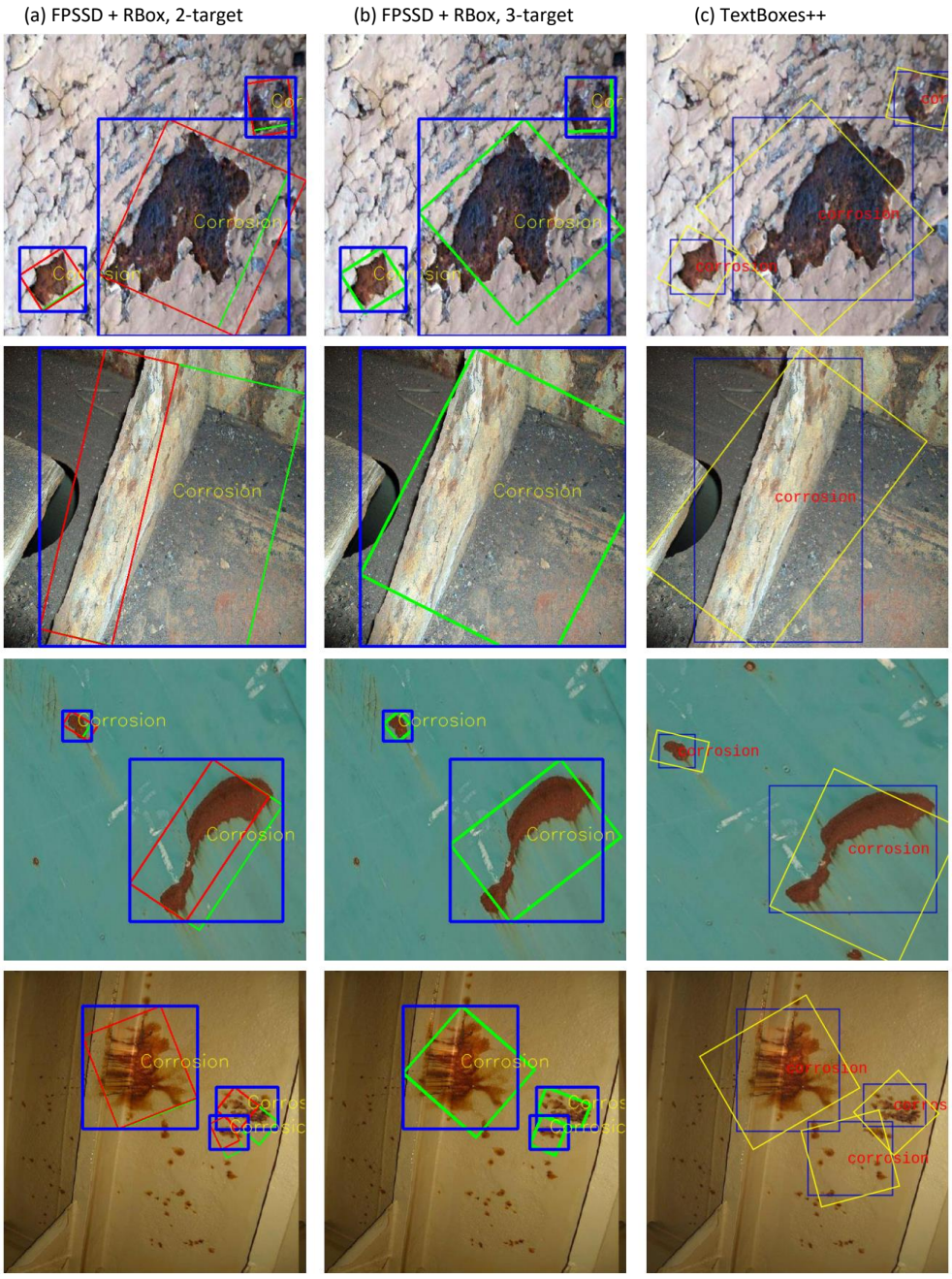


Figure 9: Examples of oriented detections for two- and three-target regression and TextBoxes++

Figure 9 shows final detection results for the full oriented-detection solution. For two-target regression, we show two oriented boxes, in red and in green, corresponding to the two possible solutions for every (d_1, d_2) pair. As already observed at a quantitative level, the RBox regression network for two-target regression gives rise to more accurate detections than TextBoxes++. Although the three-target regression approach gives rise to a single solution, it is not as accurate as the two-target variant. As for TextBoxes++,

its performance has neither resulted to be above the other approaches (despite the network has effectively converged while being fine-tuned for the two datasets).

III. Semantic segmentation-based approach

Image segmentation is a classical problem in computer vision aiming at distinguishing meaningful units in processed images. To this end, image pixels are grouped into regions that on many occasions are expected to correspond to the scene object projections. One step further identifies each unit as belonging to a particular class among a set of object classes to be recognised, giving rise to the Multi-Class Semantic Segmentation (MCSS) problem. From classical methods (e.g. region growing (Gonzalez & Woods, 2018)) to more robust methods (e.g. level-set (Wang, Ma, & Zhu, 2021) and graph-cut (Boykov & Funka-Lea, 2006)), various techniques have been proposed to achieve automatic image segmentation in a wide range of problems. Nevertheless, it has not been until recently that the performance of image segmentation algorithms has attained truly competitive levels, and this has been mostly thanks to the power of machine learning-based methodologies.

Regarding DCNN-based image segmentation, (Guo, Liu, Georgiou, & Lew, 2018) distinguish among three categories of MCSS approaches in accordance to the methodology adopted while dealing with the input images (and correspondingly the required network structure): region-based semantic segmentation, semantic segmentation based on Fully Convolutional Networks (FCN) and Weakly-Supervised semantic segmentation (WSSS). While the former follows a *segmentation using recognition* pipeline, which first detects free-form image regions, and next describes and classifies them, the second approach adopts a pixel-to-pixel map learning strategy as key idea without resorting to the image region concept, and, lastly, WSSS methods focus on achieving a performance level similar to that of Fully-Supervised methods (FSSS) but with a weaker labelling of the training image set, i.e. less spatially-informative annotations than the pixel level, to simplify the generation of ground truth data. It is true that powerful interactive tools have been developed for annotating images at the pixel level, which, in particular, just require that the annotator draws a minimal polygon surrounding the targets (see e.g. the open annotation tool by the MIT (Wada, 2016)). However, it still takes a few minutes on average to label the target areas for every picture (e.g. around 10 minutes on average for MS COCO labellers, as described by (Lin, et al., 2014)), which makes WSSS methods interesting by themselves and actually quite convenient in general. In this section, we focus on this last class of methods and propose a novel WSSS strategy based on a new loss function combining several terms to counteract the simplicity of the annotations.

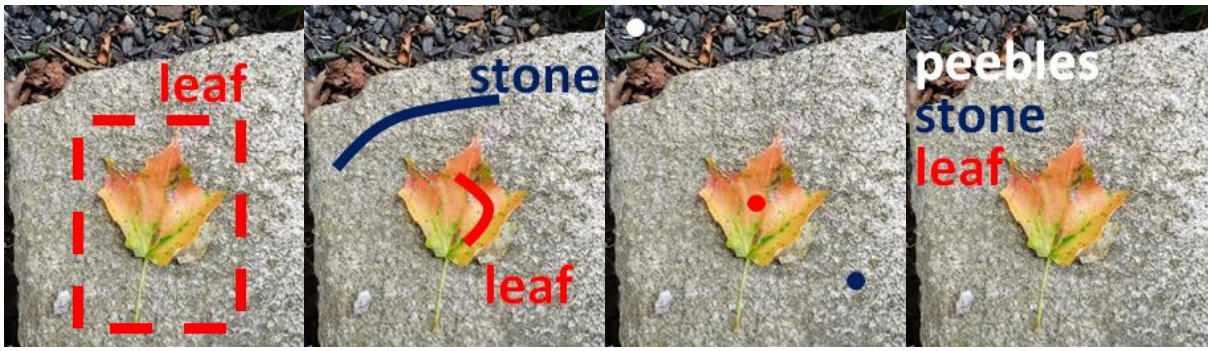
WSSS methods are characterised, among others, by the sort of weak annotation that is assumed. In this regard, (Chan, Hosseini, & Plataniotis, 2020) highlight several weak annotation methodologies, namely bounding boxes, scribbles, image points and image-level labels (see Figure 10 for an illustration of all of them). In this work, we adopt a scribble-based methodology from which training masks are derived to propagate the category information from the labelled pixels to the unlabelled pixels during network training.

The main contributions of this work are summarised as follows:

- A new loss function L comprising several partial cross entropy terms is developed to account for the vagueness of the annotations and the inherent noise of the training masks that are derived

from them. This function includes a class centroids-based loss term, named as the *Centroid Loss*, which induces a clustering process within the semantic segmentation approach.

- Another term of L is defined through a *Mean Squared Error* (MSE) loss that cooperates with the other partial cross-entropy losses to refine the segmentation results.
- The Centroid Loss is embedded over a particular implementation of Attention U-Net (Oktay, et al., 2018).



(a) (b) (c) (d)

Figure 10: Examples of weak annotations, from more to less informative: (a) bounding boxes, (b) scribbles, (c) point-level labels, (d) image-level labels.

The rest of this section is organised as follows: Section 3.1 describes the weakly-supervised methodology developed in this work; Section 3.2 discusses on the weak annotations adopted; Section 3.3 details the architecture of the network; Sections 3.4, 3.5 and 3.6 presents the different terms of the loss function; finally, Section 3.7 reports on the results of a number of experiments aiming at showing the performance of our approach from different points of view.

3.1. Methodology

Figure 11(a) illustrates fully supervised semantic segmentation approaches based on DCNN, which, applying a pixel-wise training strategy, try to make network predictions resemble the full labelling as much as possible, thus achieving good segmentation performance levels in general. By design, this kind of approach ignores the fact that pixels of the same category tend to be similar to their adjacent pixels. This similarity can, however, be exploited when addressing the WSSS problem by propagating the known pixel categories towards unlabelled pixels. In this respect, several works reliant on pixel-similarity to train the WSSS network can be found in the literature: e.g. a dense *Conditional Random Field* (CRF) is used in (Papandreou, Chen, Murphy, & Yuille, 2015), the GraphCut approach is adopted in (Zhao, Liang, & Wei, 2018), and superpixels are used in ScribbleSup (Lin, Dai, Jia, He, & Sun, 2016).

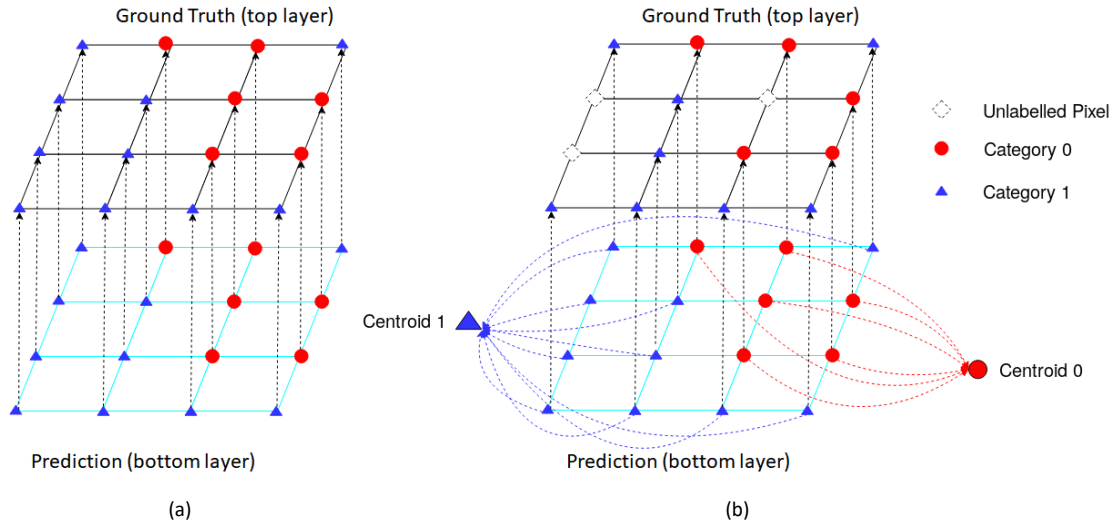
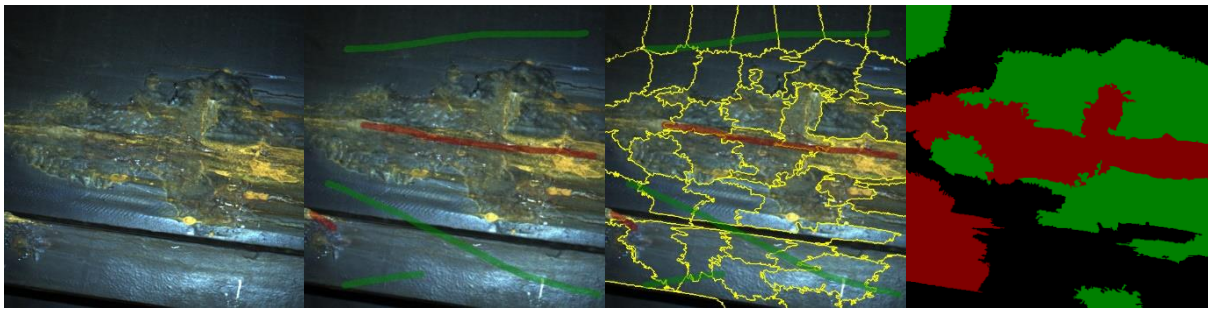


Figure 11: Illustration of (a) full supervision and (b) the weakly-supervised approach for semantic segmentation proposed in this work: (a) all pixels are labelled to make the prediction [bottom layer of the drawing] resemble the ground truth [top layer of the drawing] as much as possible after pixel-wise training; (b) to solve the WSSS problem, the category information from the incomplete ground truth, i.e. the weak annotations, is propagated towards the rest of pixels making use of pixel similarity and minimizing distances to class centroids derived from the weak annotations.

Inspired by the aforementioned, in this section, we propose a semantic segmentation approach using scribble annotations and a specific loss function intended to compensate for missing labels and errors in the training masks. To this end, class centroids determined from pixels coinciding with the scribbles, whose labelling is actually the ground truth of the problem, are used in the loss function to guide the training of the network so as to obtain improved segmentation outcomes. The process is illustrated in **Figure 11**(b).

Furthermore, similarly to ScribbleSup (Lin, Dai, Jia, He, & Sun, 2016), we also combine superpixels and scribble annotations to propagate category information and generate pseudo-masks as segmentation proposals, thus making the network converge fast and achieve competitive performance. By way of example, **Figure 12**(b) and (c) show, respectively, the scribble annotations and the superpixels-based segmentations obtained for one of the images of the training dataset. The corresponding pseudo-masks, containing more annotated pixels than the scribbles, are shown in **Figure 12**(d). As can be observed, not all pixels of the pseudo-masks are correctly labelled, which may affect segmentation performance. It is because of this fact that we incorporate the *Centroid Loss* and a *normalized MSE* terms into the full loss function.



(a) (b) (c) (d)

Figure 12: Weak annotation and propagation example: (a) original images; (b) scribbles superimposed over the original image; (c) scribbles superimposed over the superpixels segmentation result; (d) resulting pseudo-masks. Regarding the scribble annotations: red and green scribbles respectively denote corrosion and background. As for the pseudo-masks: red, black and green pixels respectively denote corrosion, background and unlabelled pixels.

The remaining methodological details are given along the next sections: we begin with the way how weak annotations are handled and how the pseudo-masks are obtained in Section 3.2, while the architecture of the network is described in Section 3.3 and the different loss terms are detailed and discussed in Sections 3.4 (partial Cross-Entropy loss, L_{pCE}), 3.5 (Centroid Loss, L_{cen}) and 3.6 (normalized MSE-term, L_{mse}, and the full loss function L).

3.2. Weak Annotations and Pseudo-Masks Generation

As already said, Figure 12(b) shows an example of scribble annotations for the visual inspection case. Because scribbles represent only a few pixels, the segmentation performance that the network can be expected to achieve will be far from satisfactory for any task that is considered. To enhance the network performance, we combine the scribbles with an oversegmentation of the image to generate pseudo-masks as segmentation proposals for training. For the oversegmentation, we make use of the Adaptive-SLIC (SLICO) algorithm (Achanta, et al., 2012), requesting enough superpixels so as not to mix different classes in the same superpixel. By way of illustration, Figure 12(c) shows an oversegmentation in 50 superpixels. Next, those pixels belonging to a superpixel that intersects with a scribble are labelled with the same class as the scribble, as shown in Figure 12(d). In Figure 12(d), the black pixels represent the background, the red pixels indicate corrosion, and the green pixels denote unlabelled pixels.

3.3. Network Architecture

In this work, we adopt U-Net (Ronneberger, Fischer, & Brox, 2015) as the base network architecture. As it is well known, U-Net evolves from the fully convolutional neural network concept and consists of a contracting path followed by an expansive path. It was developed for biomedical image segmentation, though it has been shown to exhibit good performance in general for natural images even for small training sets. Furthermore, we also embed Attention Gates (AG) in U-Net, similarly to Attention U-Net (AUN) (Oktay, et al., 2018). These attention modules have been widely used in Natural Language Processing (NLP), e.g. (Clark, Khandelwal, Levy, & Manning, 2019). Other works related with image segmentation, e.g. (Sinha & Dolz, 2021), have introduced them for enhanced performance. In our case, AGs are integrated into the decoding part of U-Net to improve its ability to segment small targets.

For completeness, we include in **Figure 13** a schematic about the operation of the AG that we make use of in this work, which, in our case, implements the series of operations described below:

$$\left(x_{i,c}^{l}\right)' = \alpha_i^l x_{i,c}^l \tag{4}$$

$$\alpha_i^l = \sigma_2\left(W_\phi^T\left(\sigma_1\left(W_x^T x_i^l + W_g^T g_i + b_g\right)\right) + b_\phi\right)$$

where the feature-map $x_i^l \in \mathbb{R}^{F_l}$ is obtained at the output of layer l for pixel i, c denotes a channel in $x_{i,c}^l$, F_l is the number of feature maps at that layer, the gating vector g_i is used for each pixel i to determine focus regions and is such that $g_i \in \mathbb{R}^{F_l}$ (after up-sampling the input from the lower layer), $W_g \in \mathbb{R}^{F_l \times 1}$, $W_x \in R^{F_l \times 1}$, and $W_\phi \in R^{1\times 1}$ are linear mappings, while $b_g \in \mathbb{R}$ and $b_\phi \in \mathbb{R}$ denote bias terms, σ_1 and σ_2 respectively represent the ReLU and the sigmoid activation functions, $\alpha_i^l \in [0,1]$ are the resulting attention coefficients, and $\Phi_{\text{att}} = \{W_g, W_x, b_g; W_\phi, b_\phi\}$ is the set of parameters of the AG.

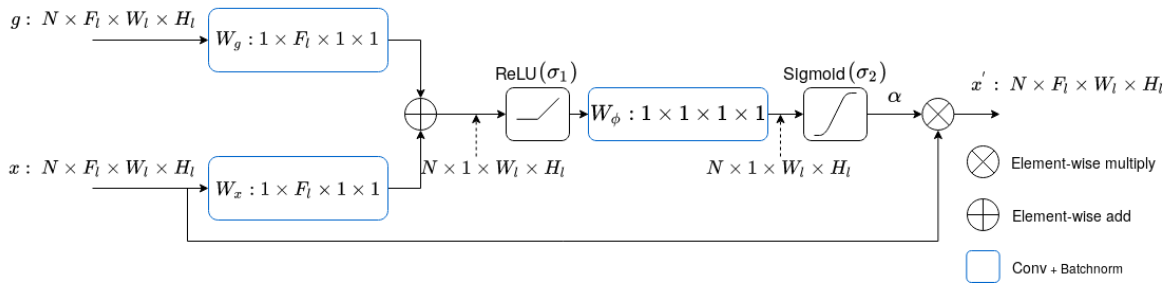


Figure 13: Schematic diagram of an Attention Gate (AG). N is the size of the mini-batch.

The attention coefficients α_i are intended to identify salient image regions and discard feature responses so as to preserve only the activations relevant to the specific task. In (Hu, Shen, & Sun, 2018), the *Squeeze-and-Excitation* (SE) block obtains attention weights in channels for filter selection. In our approach, the AGs involved calculate attention weights at the spatial level.

As shown in **Figure 14**, AGs are fed by two input tensors, one from the encoder side of U-Net and the other from the decoder side, respectively x and g in **Figure 13**. With the AG approach, spatial regions are selected on the basis of both the activations x and the contextual information provided by the gating signal g which is collected from a coarser scale. The contextual information carried by the gating vector g is hence used to highlight salient features that are passed through the skip connections. In our case, g enters the AG after an up-sampling operation that makes g and x have compatible shapes (see **Figure 13**).

Apart from the particularities of the AG that we use, which have been described above, another difference with the original AUN is the sub-network that we attach to the main segmentation network, as can be observed from the network architecture that is shown in **Figure 14**. This sub-network is intended to predict class centroids on the basis of the scribbles that are available in the image, with the aim of improving the training of the main network from the possibly noisy pseudo-masks, and hence achieve a higher level of segmentation performance. Consequently, during training: (1) our network handles two sorts of ground truth, namely scribble annotations Y_{scr} to train the attached sub-network for proper centroid predictions, and the pseudo-masks Y_{seg} for image segmentation; and (2) the augmented network yields two outputs, a set of centroids P_{cen} and the segmentation of the image P_{seg} (while during inference only the segmentation output P_{seg} is relevant). Predicted cluster centroids contribute to the Centroid Loss term L_{cen} (described in Section 3.5) of the full loss function L, which comprises two more terms (as described in Section 3.6). Thanks to the design of L, the full network –i.e. the AUN for semantic segmentation and the sub-net for centroids prediction– is trained through a joint training strategy following an end-to-end learning model. During training, the optimization of L_{cen} induces updates in the main network weights via back-propagation that are intended to lead to enhanced training and therefore produce better segmentations.

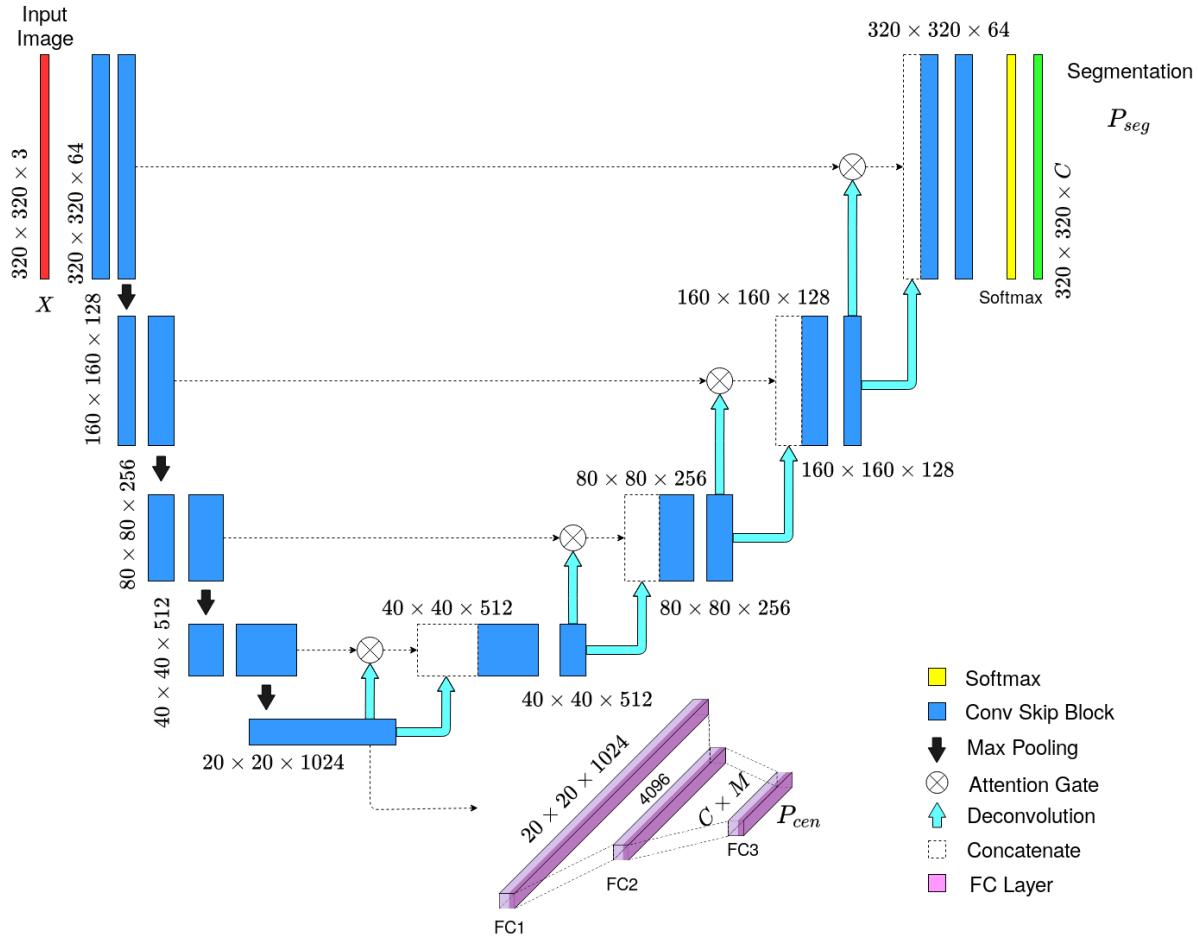


Figure 14: Block diagram of the Centroids AUN model. Size decreases gradually by a factor of 2 at each scale in the encoding part and increases by the same factor in the decoding part. In the latter, AGs are used to help the network focus on the areas of high-response in the feature maps. The *Conv Skip* block is the *skip connection* of ResNet (He, Zhang, Ren, & Sun, 2016). The sub-network at the lower part of the diagram is intended to predict class centroids. In the drawing, C denotes the number of classes and M is the dimension of the class centroids.

As can be observed, the centroids prediction sub-net is embedded into the intermediate part of the network, being fed by the last layer of the encoder side of our AUN. As shown in **Figure 14**, this sub-net consists of three blocks, each of which comprises a fully connected layer, a batch-normalization layer, and a third layer of ReLU activation functions. The shape of P_{cen} is $C \times M$, where C is the number of categories and M denotes the dimension of the feature space where the class centroids are defined. In our approach, centroid features are defined from the softmax layer of the AUN, and hence comprises C components, though we foresee to combine them with K additional features from the classes, which are incorporated externally to the operation of the network, and hence $M = C + K$. On the other side, the shape of P_{seg} is $C \times W \times H$, where (H, W) is the size of the input image.

3.4. Partial Cross-Entropy Loss

Given a C-class problem and a training set Ω, comprising a subset Ω_{L} of labelled pixels and a subset Ω_{U} of unlabelled pixels, the Partial Cross-Entropy Loss L_{pCE}, widely used for WSSS, computes the cross-entropy only for labelled pixels $p \in \Omega_{\text{L}}$, ignoring $p \in \Omega_{\text{U}}$:

$$L_{\text{pCE}} = \sum_{c=1}^{C} \sum_{p \in \Omega_L^{(1)}} - y_{g(p),c} \log y_{s(p),c}$$

(5)

where $y_{g(p),c} \in 0,1$ and $y_{s(p),c} \in [0,1]$ represent respectively the ground truth and the segmentation output. In our case, and for L_{pCE}, $\Omega_{\text{L}}^{(1)}$ is defined as the pixels labelled in the pseudo-masks (hence, pixels from superpixels not intersecting with any scribble belong to Ω_{U} and are not used by the previous loss term. Hence, $y_{g(p),c}$ refers to the pseudo-masks, i.e. Y_{seg}, while $y_{s(p),c}$ is the prediction, i.e. P_{seg}, as supplied by the *softmax* final network layer.

3.5. Centroid Loss

As can be easily foreseen, when the network is trained using the pseudo-masks, the segmentation performance depends on how accurate the pseudo-masks are and hence on the quality of superpixels, i.e. how they adhere to object boundaries and avoid mixing classes. The *Centroid Loss* function is introduced in this section for the purpose of compensating a dependence of this kind and improving the quality of the segmentation output.

In more detail, we express the *Centroid Loss* term L_{cen} as another partial cross-entropy loss:

$$L_{\text{cen}} = \sum_{c=1}^{C} \sum_{p \in \Omega_L^{(2)}} - y^*_{g(p),c} \log y^*_{s(p),c}$$

(6)

defining in this case:

- $\Omega_L^{(2)}$ as the set of pixels coinciding with the scribbles,
- $y^*_{g(p),c}$ as the corresponding labelling, and

$$\begin{aligned} y^*_{s(p),c} &= \frac{\exp(-d_{p,c})}{\sum_{c'=1}^{C} \exp(-d_{p,c'})} \\ d_{p,c} &= \frac{\left|\left|f_p - \mu_c\right|\right|_2^2}{\sum_{c'=1}^{C} \left|\left|f_p - \mu_{c'}\right|\right|_2^2} \end{aligned}$$

(7)

where: (1) f_p is the feature vector associated to pixel p and (2) μ_c denotes the centroid predicted for class c, i.e. $\mu_c \in P_{\text{cen}}$. f_p is built from the section of the softmax layer of the main network corresponding to pixel p, though f_p can be extended with the incorporation of additional external features, as already mentioned. This link between L_{pCE} and L_{cen} through the softmax layer makes both terms decrease through the joint optimization, in the sense that for a reduction in L_{cen} to take place, and hence in the full loss L, also L_{pCE} has to decrease by better predicting the class of the pixels involved. The additional features that can be incorporated in f_p try to introduce information from the classes, e.g. predominant colour, to guide even more the optimization.

In practice, this loss term *pushes* pixel class predictions towards, ideally, a subset of the corners of the C-dimensional hypercube, in accordance with the scribbles, i.e. the available ground truth. Some similarity can be certainly established with the K-means algorithm. Briefly speaking, K-means iteratively calculates a set of centroids for the considered number of clusters/classes, and associates the samples to the closest cluster in feature space, thus minimizing the intra-class variance until convergence. Some DCNN-based clustering approaches reformulate K-means as a neural network optimizing the intra-class variance loss by means of a back-propagation-style scheme (Peng, Tsang, Zhou, & Zhu, 2018). Differently from the latter, in this work, L_{cen} reformulates the unsupervised process of minimizing the distances from samples to centroids into a supervised process since the clustering takes place around the true classes defined by the labelling of the scribbles $y^*_{g(p),c}$ and the extra information that may be incorporated.

3.6. Full Loss Function

Since L_{pCE} applies only to pixels labelled in the pseudo-mask and L_{cen} is also restricted to a subset of image pixels, namely the pixels coinciding with the scribbles, we add a third loss term in the form of a normalized MSE loss L_{mse} to behave as a regularization term that involves all pixels for which a class label must be predicted $\Omega_L^{(3)}$, i.e. the full image. This term calculates the normalized distances between the segmentation result for every pixel and its corresponding centroid:

$$L_{\text{mse}} = \frac{\sum_{p \in \Omega_L^{(3)}} d_{p,c(p)}}{\left|\Omega_L^{(3)}\right|} \tag{8}$$

where $|\mathcal{A}|$ stands for the cardinality of set $\mathcal{A}$, and $d_{p,c(p)}$ is as defined by equation (7), with $c(p)$ as the class prediction for pixel p (and $\mu_{c(p)}$ the corresponding predicted centroid), taken from the softmax layer.

Finally, the complete loss function is given by

$$L = L_{\text{pCE}} + \lambda_{\text{cen}} L_{\text{cen}} + \lambda_{\text{mse}} L_{\text{mse}} \tag{9}$$

where λ_{cen} and λ_{mse} are trade-off constants.

3.7. Experimental Results and Discussion

In this section, we report on the results obtained for our WSSS approach. For a start, Sections 3.7.1 - 3.7.3 describe the experimental setup. Next, in Section 3.7.4 we discuss about the feature space where the Centroid Loss is defined and about its relationship with the weak annotations, while Section 3.7.5 evaluates the effect on the segmentation performance of several combinations of the terms of the loss function L, and Section 3.7.6 analyses the impact of weak annotations and their propagation. Subsequently, our approach is compared against two previously proposed methods in Section 3.7.7. To finish, we address final tuning and show segmentation results, for qualitative evaluation purposes, for some images of both application cases in Section 3.7.8.

3.7.1. Experimental setup

For a start, the dataset employed for training comprises images from different vessels taken under different illumination conditions, viewpoints, etc. Besides, it has been augmented with rotations and scaled versions of the original images together with random croppings, to increase the diversity of the set. Finally, as already explained, the ground truth comprises scribbles and pseudo-masks (generated in accordance to the process described in Section 3.2). By way of illustration, **Figure 15** shows some examples of weak annotations with different settings as for the width of the scribbles and the number of superpixels used for generating the pseudo-masks.

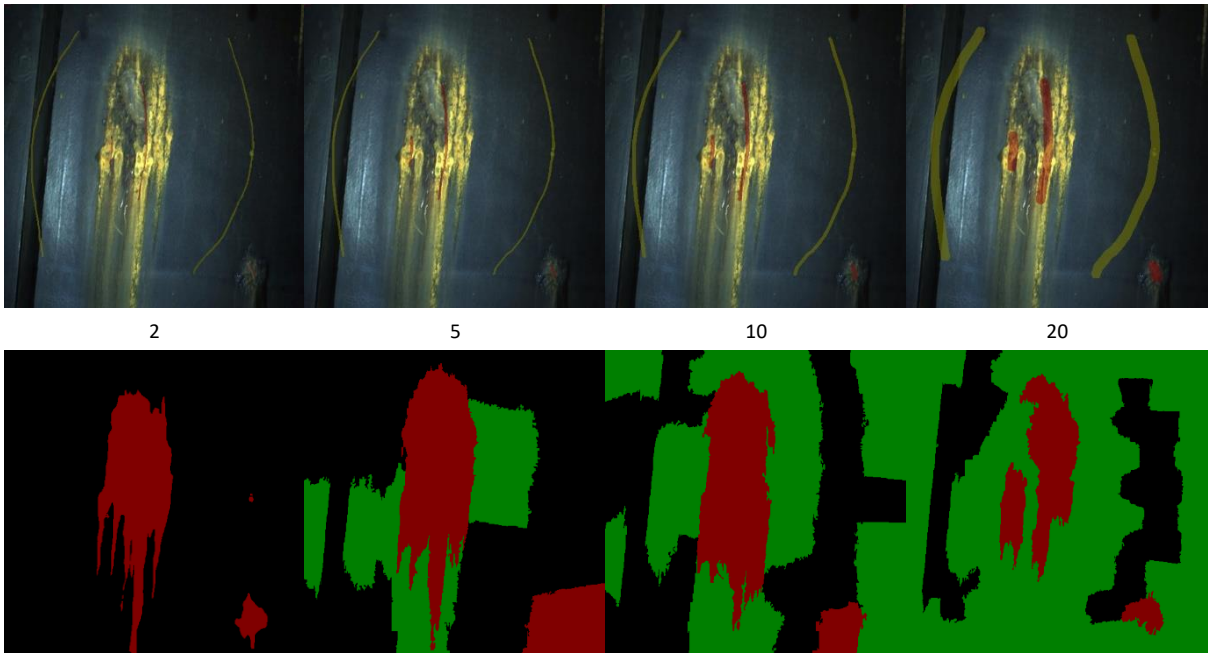


Figure 15: Examples of weak annotations and their propagation: (1st row) examples of scribble annotations of different widths, namely, from left to right, 2, 5, 10 and 20 pixels; (2nd row) the leftmost image shows the fully supervised ground truth, while the remaining images are examples of pseudo-masks generated from 20-pixel scribbles and for different amounts of superpixels, namely 30, 50, and 80, for the image of Figure 12 and, hence, for the visual inspection and the quality control application cases.
(The colour code is the same as for Figure 12.)

As well as for the object detection approach described in Section II, all experiments have been performed on a PC fitted with an NVIDIA GeForce RTX 2080 Ti GPU, a 2.9GHz 12-core CPU (Intel i9-9900K) with 32 GB RAM, and Ubuntu 64-bit. The batch size has been 8 for all experiments and the size of the input image has been 320×320 pixels, since this has turned out to be the best configuration for the aforementioned GPU.

As already mentioned, the AUN for semantic segmentation and the sub-net for centroid prediction are jointly trained following an end-to-end learning model. The network weights are initialized by means of the Kaiming method (He, Zhang, Ren, & Sun, 2015), and they are updated using a 10^{-4} learning rate for 200 epochs using the ADAM optimizer.

Best results have been obtained for the balance parameters λ_{cen} and λ_{mse} set to 1.

3.7.2. Evaluation metrics

For quantitative evaluation of our approach, we consider the following metrics:

- The mean Intersection Over Union (mIOU), which can be formally stated as follows (see e.g. (Long, Shelhamer, & Darrell, 2015)):

  Given n_{ij} as the number of pixels of class i that fall into class j, for a total of C different classes

$$\text{mIOU} = \frac{1}{C}\sum_i \frac{n_{ii}}{\sum_j n_{ij} + \sum_j n_{ji} - n_{ii}}.$$

(10)

- The mean Recall and mean Precision are also calculated to evaluate the segmentation performance for all classes. True Positive (TP), False Positive (FP) and False Negative (FN) samples are determined from the segmentation results and the ground truth. Using a macro-averaging approach (Zhang & Zhou, 2014), the mean Recall (mRec) and mean Precision (mPrec) are expressed as follows:

$$\begin{aligned}\text{mRec} &= \frac{1}{C}\left(\sum_i \frac{TP_i}{TP_i + FN_i}\right) = \frac{1}{C}\left(\sum_i \frac{TP_i}{T_i}\right)\\ \text{mPrec} &= \frac{1}{C}\left(\sum_i \frac{TP_i}{TP_i + FP_i}\right) = \frac{1}{C}\left(\sum_i \frac{TP_i}{P_i}\right)\end{aligned}$$

(11)

  where TP_i, FP_i and FN_i are, respectively, the true positives, false positives and false negatives for class i, and T_i and P_i are, respectively, the number of positives in the ground truth and the number of positive predictions, both for class i. From now on, to shorten the notation, when we refer to precision and recall, it must be understood that we are actually referring to mean precision and mean recall.

- The F_1 score as the harmonic mean of precision and recall:

$$F_1 = \frac{2 \cdot \text{mPrec} \cdot \text{mRec}}{\text{mPrec} + \text{mRec}}$$

(12)

In all experiments, we make use of fully supervised masks/ground truth for both datasets in order to be able to report quantitative measurements about the segmentation performance. This ground truth has been manually generated only for this purpose, it has been used for training only when referring to the performance of the fully-supervised approach, for comparison purposes between the fully- and weakly-supervised solutions.

To finish, in a number of experiments we also report on the quality of the pseudo-masks, so that the segmentation performance reported can be correctly valued. To this end, we calculate a weak mIOU (wmIOU) using equation (10) between the pseudo- and the fully-supervised masks involved.

3.7.3. Overall view of the experiments

The experiments that are going to be discussed along the next sections consider different configurations for the different elements that are involved in our semantic segmentation approach. These configurations, which are enumerated in **Table 5**, involve:

- different widths of the scribble annotations used as ground truth, namely 2, 5, 10 and 20 pixels,
- different amounts of superpixels for generating the pseudo-masks, namely 30, 50 and 80,
- two ways of defining the feature space for the class centroids: from exclusively the *softmax* layer of AUN and combining those features with other features from the classes.

Table 5: Labels for the different experiments performed, varying the width of scribbles, the number of superpixels employed for generating the pseudo-masks, and the terms involved in the loss function employed during training. SMX stands for *softmax*.

Configuration	Label	Scribbles width	Number of superpixels	Centroid features	Supervision	Loss function
lower baseline (G1)	E-SCR2	2	-	-	only scribbles	L_{pCE}
	E-SCR5	5	-	-		
	E-SCR10	10	-	-		
	E-SCR20	20	-	-		
	E-SCR20-SUP30	20	30	-	pseudo-masks	L_{pCE}
	E-SCR20-SUP50	20	50	-		
	E-SCR20-SUP80	20	80	-		
G2 / G3	E-SCR2-N	2	-	SMX	only scribbles	$L_{\text{pCE}} + L_{\text{cen}}\ [+L_{\text{mse}}]$
	E-SCR2-NRGB	2	-	SMX & norm. RGB		
	E-SCR5-N	5	-	SMX		
	E-SCR5-NRGB	5	-	SMX & norm. RGB		
	E-SCR10-N	10	-	SMX		
	E-SCR10-NRGB	10	-	SMX & norm. RGB		
	E-SCR20-N	20	-	SMX		
	E-SCR20-NRGB	20	-	SMX & norm. RGB		
	E-SCR20-SUP30-N	20	30	SMX	pseudo-masks	$L_{\text{pCE}} + L_{\text{cen}}\ [+L_{\text{mse}}]$
	E-SCR20-SUP30-NRGB	20	30	SMX & norm. RGB		
	E-SCR20-SUP50-N	20	50	SMX		
	E-SCR20-SUP50-NRGB	20	50	SMX & norm. RGB		
	E-SCR20-SUP80-N	20	80	SMX		
	E-SCR20-SUP80-NRGB	20	80	SMX & norm. RGB		
upper baseline	E-FULL	-	-	-	full mask	L_{CE}

Notice that the first rows of **Table 5** refer to experiments where the loss function used for training is just the partial cross-entropy, as described in equation (5), and therefore can be taken as a lower baseline method. The upper baseline would correspond to the configuration using full masks and the cross entropy loss L_{CE} for training, i.e. full supervised semantic segmentation, which can also be found in **Table 5** as the last row.

Apart from the aforementioned variations, we also analyse the effect of several combinations of the loss function terms, as described in equation (9), defining three groups of experiments: Group 1 (G1), which indicates that the network is trained by means of only L_{pCE}, and hence would also coincide with the lower baseline; Group 2 (G2), which denotes that the network is trained by means of the combination of L_{pCE} and L_{cen}; and Group 3 (G3), for which the network is trained using the full loss function as described in equation (9).

Finally, we compare our segmentation approach with two other alternative approaches also aimed at solving the WSSS problem through a modified loss function. These loss functions are the *Constrained-size Loss* (L_{size}) (Kervadec, et al., 2019) and the *Seed, Expand, and Constrain* (SEC) *Loss* (L_{sec}) (Kolesnikov & Lampert, 2016):

$$\begin{aligned} L_{\text{size}} &= L_{\text{pCE}} + \lambda_{\text{size}} L_{\mathcal{C}(V_S)} \\ L_{\text{sec}} &= L_{\text{seed}} + L_{\text{expand}} + L_{\text{constrain}} \end{aligned} \quad (13)$$

On the one hand, λ_{size} for the $L_{\mathcal{C}(V_S)}$ term is set to 10^{-3}. On the other hand, regarding L_{sec}, it consists of three terms, the seed loss L_{seed}, the expand loss L_{expand}, and the constrain loss $L_{\text{constrain}}$. In our case, we feed L_{seed} from the scribble annotations, while, regarding L_{expand} and $L_{\text{constrain}}$, we adopt the same configuration as in the original work.

3.7.4. About the centroid loss feature space and the weak annotations

Given the relevance that colour features can have in image semantic segmentation performance (Liu, Deng, & Yang, 2018), the experiments reported in this section consider the incorporation of colour data from the classes into the calculation and minimization of the Centroid and the MSE loss functions, L_{cen} and L_{mse}. More specifically, we adopt a simple strategy by making use of normalized RGB features [1]:

$$\text{nRGB}_p = \frac{1}{R_p + G_p + B_p}\left(R_p, G_p, B_p\right) \quad (14)$$

As mentioned in Section 3.3, the shape of P_{cen} is $C \times M$, where $M = C + K$, and K is the number of additional features from the classes that we incorporate into the network optimization problem. Therefore, in our case, $K = 3$. Of course, more sophisticated hand-crafted features can be incorporated into the process, though the idea of this experiment has been to make use of simple features.

Table 6 and **Table 7** evaluate the performance of our approach for different combinations of loss terms, for the two centroid feature spaces outlined before, and also depending on the kind of weak annotation that is employed as ground truth and their main feature value, i.e. width for scribbles and number of superpixels for pseudo-masks. Besides, we consider two possibilities of producing the final labelling: from the output of the segmentation network and from the clustering deriving from the predicted class centroids, i.e. label each pixel with the class label of the closest centroid; from now on, to simplify the discussion despite the language abuse, we will refer to the latter kind of output as that resulting from *clustering*.

As can be observed in **Table 6**, segmentation and clustering mIOU for experiments E-SCR*-NRGB is lower than the mIOU for experiments E-SCR*-N, with a large gap in performance in a number of cases, what suggests that the RGB features actually do not contribute —rather the opposite— on improving segmentation performance when scribble annotations alone are used as supervision information for the visual inspection dataset.

[1] If $R_p = G_p = B_p = 0$, then $\text{nRGB}_p = (0,0,0)$.

As for **Table 7**, contrary to the results shown in **Table 6**, the performance that can be observed from experiments E-SCR20-SUP*-NRGB results to be similar to that of experiments E-SCR20-SUP*-N. Additionally, the mIOU of some experiments where the integrated features, i.e. *softmax* and colour, are used is even higher than if only the *softmax* features are used (e.g. E-SCR20-SUP80-N/NRGB, sixth row of **Table 7**).

At a global level, both **Table 6** and **Table 7** show that our approach requires a higher number of labelled pixels to achieve higher segmentation performance when the integrated features are employed. In contrast, for the visual inspection task, the use of *softmax* features only requires the scribble annotations to achieve good performance. Nevertheless, our approach using *softmax* features achieves higher mIOU than using the integrated features in most of the experiments. As a consequence, only *softmax* features are involved in the next experiments.

Table 6: Segmentation performance for different centroid feature spaces and different widths of the scribble annotations. **N* denotes that only the SMX (*softmax*) features are used to compute L_{cen} and L_{mse}, while **NRGB* denotes that the feature space for centroids prediction comprises both SMX and RGB features. *Seg* denotes that the segmentation output comes directly from the segmentation network, while *Clu* denotes that the segmentation output is obtained from clustering.

Experiments	wmIOU	L_{pCE}	L_{cen}	L_{mse}	mIOU (Seg)	mIOU (Seg,*N)	mIOU (Seg,*NRGB)	mIOU (Clu,*N)	mIOU (Clu,*NRGB)
E-SCR2	0.2721	✓			0.3733	-	-	-	-
E-SCR5	0.2902	✓			0.4621	-	-	-	-
E-SCR10	0.3074	✓			0.4711	-	-	-	-
E-SCR20	0.3233	✓			0.5286	-	-	-	-
E-SCR2-*	0.2721	✓	✓		-	0.6851	0.4729	0.6758	0.3889
E-SCR5-*	0.2902	✓	✓		-	0.6798	0.4989	0.6706	0.6020
E-SCR10-*	0.3074	✓	✓		-	0.6992	0.5130	0.6710	0.6267
E-SCR20-*	0.3233	✓	✓		-	0.6852	0.5562	0.6741	0.6164
E-SCR2-*	0.2721	✓	✓	✓	-	0.6995	0.4724	0.6828	0.3274
E-SCR5-*	0.2902	✓	✓	✓	-	0.7134	0.4772	0.7001	0.2982
E-SCR10-*	0.3074	✓	✓	✓	-	0.7047	0.4796	0.6817	0.3130
E-SCR20-*	0.3233	✓	✓	✓	-	0.6904	0.5075	0.6894	0.6187

Table 7: Segmentation performance for different centroid feature spaces and for different amounts of superpixels to generate the pseudo-masks. **N* denotes that only the SMX (*softmax*) features are used to compute L_{cen} and L_{mse}, while **NRGB* denotes that the feature space comprises both SMX and RGB features. *Seg* denotes that the segmentation output comes directly from the segmentation network, while *Clu* denotes that the segmentation output is obtained from clustering.

Experiments	wmIOU	L_{pCE}	L_{cen}	L_{mse}	mIOU (Seg)	mIOU (Seg,*N)	mIOU (Seg,*NRGB)	mIOU (Clu,*N)	mIOU (Clu,*NRGB)
E-SCR20-SUP30	0.6272	✓			0.6613	-	-	-	-
E-SCR20-SUP50	0.6431	✓			0.7133	-	-	-	-
E-SCR20-SUP80	0.6311	✓			0.7017	-	-	-	-
E-SCR20-SUP30-*	0.6272	✓	✓		-	0.6848	0.6847	0.7081	0.6859
E-SCR20-SUP50-*	0.6431	✓	✓		-	0.7447	0.7368	0.7372	0.7136
E-SCR20-SUP80-*	0.6311	✓	✓		-	0.7242	0.7355	0.7127	0.6761
E-SCR20-SUP30-*	0.6272	✓	✓	✓	-	0.6919	0.7071	0.6987	0.7076
E-SCR20-SUP50-*	0.6431	✓	✓	✓	-	0.7542	0.7133	0.7491	0.7294
E-SCR20-SUP80-*	0.6311	✓	✓	✓	-	0.7294	0.7246	0.7268	0.7118

3.7.5. Effect of the loss function terms

This section considers the effect of L_{cen} and L_{mse} on the segmentation results by analysing the performance achieved in the experiments of groups G1, G2 and G3. From **Table 6**, one can see that the mIOU of experiments in G2 is significantly higher than that of experiments in G1, where the maximum gap in mIOU between G1 and G2 is 0.3118 (E-SCR2 and E-SCR2-N). As for the segmentation performance for G3 experiments, it is systematically above that of G2 experiments for the same width of the scribble annotations and if centroids are built only from the *softmax* features. When the colour data is incorporated, segmentation performance decreases from G2 to G3.

Table 7 also shows performance improvements from G2 experiments, i.e. when L_{cen} is incorporated into the loss function, over the performance observed from experiments in G1, and, in turn, segmentation results from G3 experiments are superior to that of G2 experiments. Therefore, the incorporation of the L_{cen} and L_{mse} terms into the loss function benefits performance, gradually increasing the mIOU of the resulting segmentations.

Regarding the segmentation computed from clustering, the mIOU of experiments in G3 is also higher than that of experiments in G2. In addition, it can be found out in **Table 6** and **Table 7** that the mIOU from clustering in some G2 experiments is slightly higher than that for G3 experiments (e.g. E-SCR20-SUP30-N), while the mIOU from segmentation in G2 is lower than that of G3. In other words, it seems that L_{mse}, in some cases, makes the segmentation quality from clustering deteriorate.

Overall, the incorporation of L_{cen} and L_{mse} improves segmentation performance and labelling from segmentation turns out to be superior to that deriving from class centroids.

3.7.6. Impact of weak annotations and their propagation

In this section, we evaluate our approach under different weak annotations and their propagation, and discuss on their impact on segmentation performance. To this end, we plot in Figure 16 the mIOU (complementarily to **Table 6** and **Table 7**), recall and precision values resulting after the supervision of different sorts of weak annotations. A first analysis of these plots reveals that the curves corresponding to the G3 experiments are above those for G1 and G2 groups for all the performance metrics considered.

Figure 16(a) shows that the mIOU values for the G2 and G3 groups are above those for G1 (the lower baseline), which follows a similar shape as the wmIOU values, while those from G2 and G3 groups keep at a more or less constant level for the different sorts of weak annotations. Globally, this behaviour clearly shows that the scribbles are enough for describing the classes in this case of binary classification problem, though the pseudo-masks (G2 and G3 groups) permits achieving a higher performance. The fact that the lower baseline (G1 group) always achieves lower mIOU values also corroborates the relevance of the Centroid Loss, despite its ultimate contribution to the segmentation performance is also affected by the quality of the weak annotations involved, i.e. the pseudo-masks deriving from scribbles and superpixels for the cases of the G2 and G3 groups.

Additionally, observing the precision curves shown in Figure 16(c), one can notice that the precision for exclusively the weak annotations show a sharp decline when the weak annotations shift from scribbles to pseudo-masks. As can be noticed from the pseudo-masks shown in the second row of **Figure 15**, when the number of superpixels is low, e.g. 30, the pseudo-masks contain an important number of incorrectly labelled pixels, significantly more than that of the scribble annotations, and this is the reason for the

aforementioned decline. The recall curves, however, exhibit an upward trend as can be observed in Figure 16(b) because of the larger amount of information ultimately provided by the weak annotations. On the other side, we can also notice that, in general, precision and recall values are higher for the G3 group than for the G2 group, and both curves are above those for the G1 group. Finally, the output from clustering does not clearly lead to a different performance, better or worse, over the alternative outcome from the segmentation network.

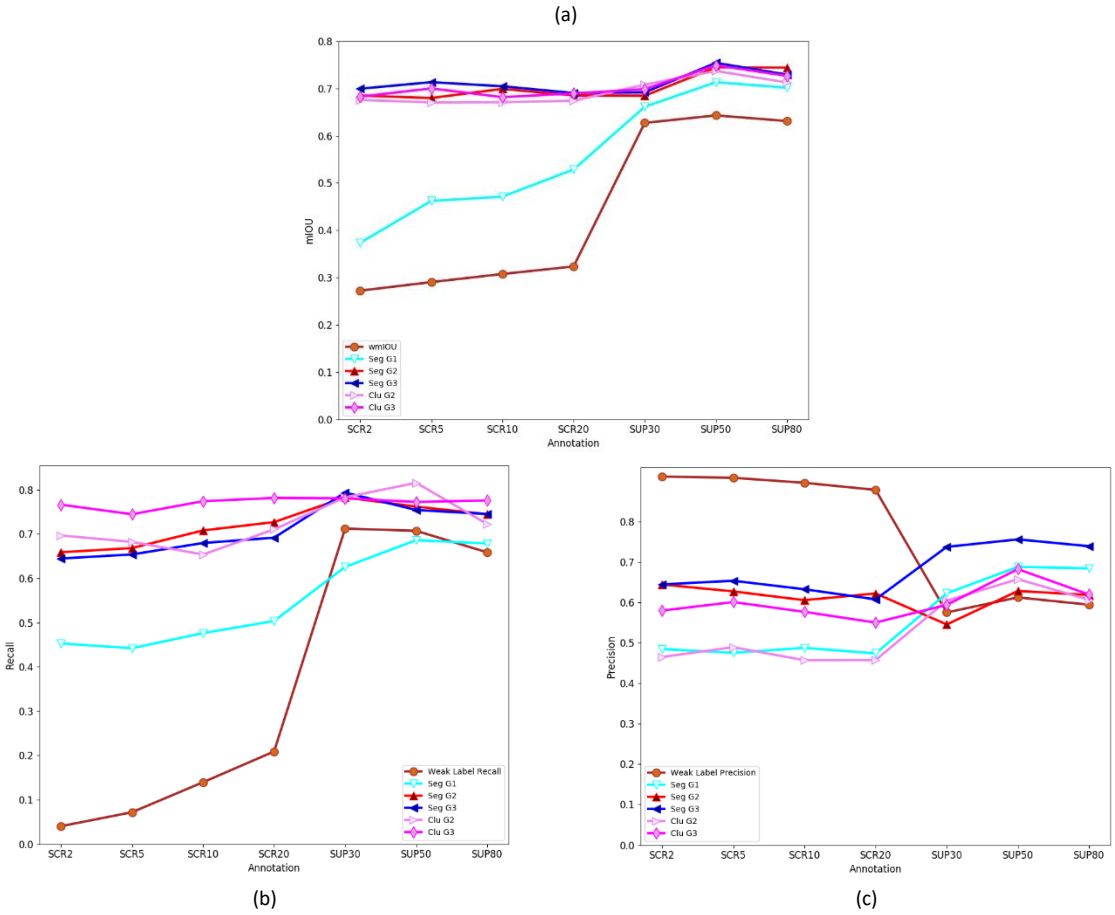


Figure 16: Performance metrics for the WSSS approach proposed in this work under different sorts of weak annotations. From left to right, the three figures plot respectively the mIOU, the mean Recall, and the mean Precision. SUP30, SUP50 and SUP80 labels correspond to the use of 20 pixel-wide scribbles.

From a global perspective, all this suggests that (a) segmentation quality benefits from the use of pseudo-masks, (b) overcoming always the lower baseline based on the use of exclusively scribbles, (c) despite the incorrectly labelled pixels contained in pseudo-masks, (d) provided that the proper loss function is adopted, e.g. the full loss expressed in equation (9), which in particular (e) comprises the Centroid Loss.

3.7.7. Comparison with other loss functions

In **Table 8**, we compare the segmentation performance of our approach with that resulting from the use of the Constrained-size Loss L_{size} and the SEC Loss L_{sec} for different variations of weak annotations. As for the visual inspection task, the network trained with L_{sec} is clearly inferior to the one resulting for our loss function, and the same can be said for L_{size}, although, in this case, the performance gap is shorter, even negligible when the width of the scribbles is of 20 pixels. When the pseudo-masks are involved, our approach is also better though the difference with both L_{size} and L_{sec} is shorter.

Summing up, we can conclude that the loss function proposed in equation (9) outperforms both the Constrained-size Loss L_{size} and the SEC Loss L_{sec} on the visual inspection task.

Table 8: Comparison of different loss functions. mIOU values are provided. Best performance is highlighted in bold.

Weak Annotation	L_{size}	L_{sec}	Ours
E-SCR2-N	0.6098	0.4366	**0.6995**
E-SCR5-N	0.6537	0.4372	**0.7134**
E-SCR10-N	0.6754	0.5486	**0.7047**
E-SCR20-N	**0.6909**	0.5624	0.6904
E-SCR20-SUP30-N	**0.7068**	0.6397	0.6919
E-SCR20-SUP50-N	0.6769	0.7428	**0.7542**
E-SCR20-SUP80-N	0.7107	0.6546	**0.7294**

3.7.8. Final tuning and results

As has been already highlighted along the previous sections, the network trained by means of the loss function described in equation (9), which in particular comprises the Centroid Loss, attains the best segmentation performance against other approaches for the visual inspection task. In order to check whether segmentation performance can increase further, in this section we incorporate a dense CRF as a post-processing stage of the outcome of the network. **Table 9** collects metric values for the final performance attained by the proposed WSSS method and as well by the upper baseline method (E-FULL). To assess the influence of the CRF-based stage, in **Table 9**, we report mIOU, precision and recall values, together with the F_1 score.

Table 9: Segmentation results for the full loss function (G3). *Seg* denotes that the segmentation output comes directly from the segmentation network, while *Clu* denotes that the segmentation output is obtained from clustering. *CRF refers to the performance (mIOU) after dense CRF post-processing. Best performance for the WSSS approach is highlighted in bold.

Experiments	wmIOU	mIOU (seg)	mRec (seg)	mPrec (seg)	F_1 (seg)	mIOU (clu)	mRec (clu)	mPrec (clu)	F_1 (clu)	*CRF (seg)
E-SCR2-N	0.2721	0.6995	0.6447	0.6452	0.6449	0.6828	0.7663	0.5803	0.6605	0.7068
E-SCR5-N	0.2902	0.7134	0.6539	0.6542	0.6540	0.7001	0.7447	0.6015	0.6655	0.7212
E-SCR10-N	0.3074	0.7047	0.6797	0.6332	0.6556	0.6817	0.7741	0.5772	0.6613	0.7241
E-SCR20-N	0.3233	0.6904	0.6917	0.6081	0.6472	0.6894	**0.7816**	0.5507	0.6461	0.7172
E-SCR20-SUP30-N	0.6272	0.6919	**0.7937**	0.7081	0.7485	0.6987	0.7806	0.5946	0.6750	0.7489
E-SCR20-SUP50-N	**0.6431**	**0.7542**	0.7543	**0.7567**	**0.7555**	**0.7491**	0.7725	**0.6830**	**0.7250**	**0.7859**
E-SCR20-SUP80-N	0.6311	0.7294	0.7452	0.7397	0.7424	0.7268	0.7758	0.6200	0.6892	0.7693
E-FULL	1.0000	0.8333	0.8537	0.9119	0.8818	-	-	-	-	0.8218

Regarding the visual inspection task, **Table 9** shows that case E-SCR20-SUP80-N leads to the best segmentation mIOU (0.7542). After the incorporation of the dense CRF, the mIOU reaches a value of 0.7859, with a performance gap with E-FULL of 0.8333 – 0.7859 = 0.0474. Case E-SCR20-SUP30-N attains the highest recall (0.7937), but the corresponding precision (0.7081) and F_1 score (0.7485) are not the highest; the mIOU is also the second lowest (0.6919). This is because the segmentation result for E-SCR20-SUP30-N contains more incorrect predictions than E-SCR20-SUP50-N. Consequently, a configuration of 20-pixel scribbles and 50 superpixels for pseudo-mask generation seem to lead to the best performance, with a slightly increase thanks to the CRF post-processing stage. The outcome from clustering is not far in quality to those values, but, as can be observed, it is not as good (the best mIOU and F_1 scores are, respectively, 0.7491 and 0.7250). Further, it must be noticed that all the aforementioned results from pseudo-masks whose wmIOU is slightly above 60%.

From a global perspective, the results obtained indicate that 20-pixel scribbles, together with a rather higher number of superpixels, so that they adhere better to object boundaries, are the best options. In comparison with the lower baseline (G1 group), the use of the full loss function, involving the Centroid Loss, clearly makes training improve segmentation performance significantly, with a slight decrease regarding full supervision. Finally, segmentation results deriving from clustering are not better.

Figure 17 shows examples of segmentation results. As can be observed, the segmentations resulting from our approach are very similar to those from the upper baseline (E-FULL). Moreover, as expected, results from clustering are basically correct though tend to label incorrectly pixels (false positives) from around correct labellings (true positives).

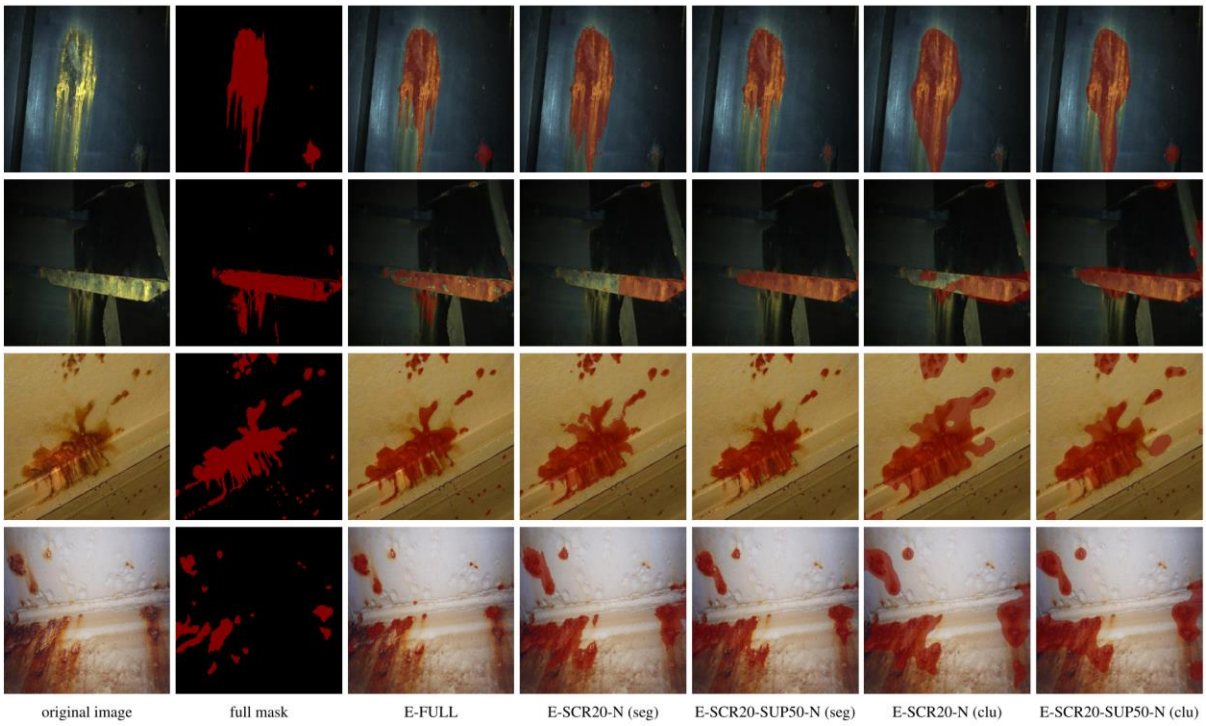


Figure 17: Examples of segmentation results: (1st column) original images, (2nd column) full mask, (3rd column) results of the fully supervised approach, (4th & 5th columns) segmentation output for E-SCR20-N and E-SCR20-SUP50-N after the use of the dense CRF, (6th & 7th columns) segmentation output from clustering for the same configurations.

Summing up, the use of the Centroid Loss has made possible training a semantic segmentation network using a small number of labelled pixels. Though the performance of the approach is inferior to that of a fully supervised approach, the resulting gap has turned out to be rather short, given the challenges arising from the use of weak annotations.

IV. Conclusions

On the one hand, this report describes a two-stage arbitrarily-oriented object detection method for regressing the parameters of oriented bounding boxes for the case of CBC detection. The first stage of our solution comprises a feature pyramid architecture that has been embedded in an SSD-like network to fuse the available feature maps, giving rise to the FPSSD network. Besides, prior boxes for un-oriented bounding box regression have been chosen on the basis of a clustering process over the available datasets. In the second stage, a simple but effective neural network has been designed to regress the parameters of oriented bounding boxes. The design process has considered two parameterizations of oriented bounding boxes, being the two-target RBox regression model the variant with highest performance. The experimental results of the whole solution show improved performance over other detection approaches.

On the other hand, this report also describes a weakly-supervised segmentation approach based on Attention U-Net. The loss function comprises three terms, namely a partial cross-entropy term, the so-called Centroid Loss and a regularization term based on the mean squared error. They all are jointly optimized using an end-to-end learning model. As has been reported in the experimental results section, for the visual inspection task, our approach can achieve competitive performance with regard to full supervision, with a reduced labelling cost to generate the necessary semantic segmentation ground truth. Under weak annotations of varying quality, our approach has been able to achieve good segmentation performance, counteracting the negative impact of the imperfect labellings employed.

The performance gap between our weakly-supervised approach and the corresponding fully-supervised approach has shown to be rather reduced regarding the mIOU values, although non-negligible, as well as for precision and recall. This suggests looking for alternatives even less sensitive to the imperfections of the ground truth deriving from the weak annotations, aiming at closing the aforementioned gap. In this regard, future work will focus on other deep backbones for semantic segmentation, e.g. DeepLab (Chen, Zhu, Papandreou, Schroff, & Adam, 2018).

V. References

Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., & Süsstrunk, S. (2012). SLIC superpixels compared to state-of-the-art superpixel methods. *IEEE Transacions on Pattern Analysis and Machine Intelligence, 34*, 2274–2282.

Boykov, Y., & Funka-Lea, G. (2006). Graph Cuts and Efficient N-D Image Segmentation. *International Journal of Computer Vision, 70*, 109-131.

Chan, L., Hosseini, M. S., & Plataniotis, K. N. (2020). A Comprehensive Analysis of Weakly-Supervised Semantic Segmentation in Different Image Domains. *International Journal of Computer Vision, 129*, 361–384.

Chen, L.-C., Zhu, Y., Papandreou, G., Schroff, F., & Adam, H. (2018). Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation. *Proc. European Conference on Computer Vision*, (pp. 833–851).

Clark, K., Khandelwal, U., Levy, O., & Manning, C. D. (2019). What Does BERT Look At? An Analysis of BERT's Attention. *Proc. ACL Workshop BlackboxNLP: Analysing and Interpreting Neural Networks for NLP*, (pp. 276-286).

Everingham, M., Eslami, S. M., Van Gool, L., Williams, C. K., Winn, J., & Zisserman, A. (2015). The Pascal Visual Object Classes Challenge: A Retrospective. *International Journal of Computer Vision, 111*, 98–136.

Fu, C.-Y., Liu, W., Ranga, A., Tyagi, A., & Berg, A. C. (2017). DSSD: Deconvolutional single shot detector. *arXiv preprint arXiv:1701.06659*. Retrieved from https://arxiv.org/abs/1701.06659

Girshick, R. (2015). Fast R-CNN. *Proc. IEEE International Conference on Computer Vision*, (pp. 1440–1448).

Gonzalez, R. C., & Woods, R. E. (2018). *Digital Image Processing.* Pearson.

Guo, Y., Liu, Y., Georgiou, T., & Lew, M. S. (2018, November). A review of semantic segmentation using deep neural networks. *International Journal of Multimedia Information Retrieval, 7*, 87–93.

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. *Proc. IEEE International Conference on Computer Vision*, (pp. 1026-1034).

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *Proc. IEEE International Conference on Computer Vision and Pattern Recognition*, (pp. 770–778).

Hu, J., Shen, L., & Sun, G. (2018). Squeeze-and-excitation networks. *Proc. IEEE International Conference on Computer Vision and Pattern Recognition*, (pp. 7132–7141).

Ioffe, S., & Szegedy, C. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing. *arXiv preprint arXiv:1502.03167*. Retrieved from http://arxiv.org/abs/1502.03167

Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., . . . Darrell, T. (2014). Caffe: Convolutional architecture for fast feature embedding. *Proc. ACM International Conference on Multimedia*, (pp. 675–678).

Kervadec, H., Dolz, J., Tang, M., Granger, E., Boykov, Y., & Ben Ayed, I. (2019). Constrained-CNN losses for weakly supervised segmentation. *Medical image analysis, 54*, 88–99.

Kolesnikov, A., & Lampert, C. H. (2016). Seed, expand and constrain: Three principles for weakly-supervised image segmentation. *Proc. European Conference on Computer Vision*, (pp. 695–711).

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. In F. P. Weinberger (Ed.), *Advances in Neural Information Processing Systems*, (pp. 1097-1105).

Lecun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE, 86*, 2278-2324.

Li, Z., & Zhou, F. (2017). FSSD: feature fusion single shot multibox detector. *arXiv preprint arXiv:1712.00960*. Obtenido de https://arxiv.org/abs/1712.00960

Liao, M., Shi, B., & Bai, X. (2018). Textboxes++: A single-shot oriented scene text detector. *IEEE Transactions on Image Processing, 27*, 3676–3690.

Lin, D., Dai, J., Jia, J., He, K., & Sun, J. (2016). ScribbleSup: Scribble-supervised convolutional networks for semantic segmentation. *Proc. IEEE International Conferece on Computer Vision and Pattern Recognition*, (pp. 3159–3167).

Lin, T.-Y., Dollár, P., Girshick, R., He, K., Hariharan, B., & Belongie, S. (2017). Feature pyramid networks for object detection. *Proc. International Conference on Computer Vision and Pattern Recognition*, (pp. 2117–2125).

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., . . . Tuytelaars, T. (2014). Microsoft COCO: Common Objects in Context. In D. Fleet, T. Pajdla, B. Schiele, & T. Tuytelaars (Ed.), *Proc. European Conference on Computer Vision*, *8693*, pp. 740–755.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., & Berg, A. C. (2016). SSD: Single shot multibox detector. *Proc. European Conference on Computer Vision*, (pp. 21–37).

Liu, X., Deng, Z., & Yang, Y. (2018). Recent progress in semantic image segmentation. *Artificial Intelligence Review, 52*, 1089–1106.

Long, J., Shelhamer, E., & Darrell, T. (2015). Fully convolutional networks for semantic segmentation. *Proc. IEEE Internationanl Conference on Computer Vision and Pattern Recognition*, (pp. 3431–3440).

Oktay, O., Schlemper, J., Le Folgoc, L., Lee, M., Heinrich, M., Misawa, K., . . . Rueckert, D. (2018). Attention U-Net: Learning where to look for the pancreas. *Proc. Conference on Medical Imaging with Deep Learning.*

Papandreou, G., Chen, L.-C., Murphy, K. P., & Yuille, A. L. (2015). Weakly- and semi-supervised learning of a deep convolutional network for semantic image segmentation. *Proc. IEEE International Conference on Computer Vision*, (pp. 1742–1750).

Peng, X., Tsang, I. W., Zhou, J. T., & Zhu, H. (2018). k-meansNet: When k-means meets differentiable programming. *arXiv preprint arXiv:1808.07292*. Retrieved from https://arxiv.org/abs/1808.07292

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. *Proc. IEEE International Conference on Computer Vision and Pattern Recognition*, (pp. 779–788).

Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster R-CNN: Towards real-time object detection with region proposal networks. *Proc. International Conference on Neural Information Processing Systems*, (págs. 91–99).

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. *Proc. International Conference on Medical Image Computing and Computer-Assisted Intervention*, (pp. 234–241).

Sinha, A., & Dolz, J. (2021). Multi-scale self-guided attention for medical image segmentation. *IEEE Journal of Biomedical and Health Informatics, 25*, 121-130.

Wada, K. (2016). labelme: Image Polygonal Annotation with Python. *labelme: Image Polygonal Annotation with Python*.

Wang, Z., Ma, B., & Zhu, Y. (2021). Review of Level Set in Image Segmentation. *Archives of Computational Methods in Engineering*, 28, pages 2429–2446.

Zhang, M.-L., & Zhou, Z.-H. (2014). A Review on Multi-Label Learning Algorithms. *IEEE Transactions on Knowledge and Data Engineering, 26*, 1819–1837.

Zhao, X., Liang, S., & Wei, Y. (2018). Pseudo mask augmented object detection. *Proc. IEEE International Conferece on Computer Vision and Pattern Recognition*, (pp. 4061–4070).