Model-Assisted Labeling via Explainability for Visual Inspection of Civil Infrastructures

Klara Janouskova[1], Mattia Rigotti[2], Ioana Giurgiu[2(✉)], and Cristiano Malossi[2]

[1] Visual Recognition Group, Faculty of Electrical Engineering, Czech Technical University in Prague, Prague, Czechia
klara.janouskova@fel.cvut.cz

[2] IBM Research Zürich, Rüschlikon, Switzerland
{mrg,igi,acm}@zurich.ibm.com

Abstract. Labeling images for visual segmentation is a time-consuming task which can be costly, particularly in application domains where labels have to be provided by specialized expert annotators, such as civil engineering. In this paper, we propose to use attribution methods to harness the valuable interactions between expert annotators and the data to be annotated in the case of defect segmentation for visual inspection of civil infrastructures. Concretely, a classifier is trained to detect defects and coupled with an attribution-based method and adversarial climbing to generate and refine segmentation masks corresponding to the classification outputs. These are used within an assisted labeling framework where the annotators can interact with them as proposal segmentation masks by deciding to accept, reject or modify them, and interactions are logged as weak labels to further refine the classifier. Applied on a real-world dataset resulting from the automated visual inspection of bridges, our proposed method is able to save more than 50% of annotators' time when compared to manual annotation of defects.

Keywords: Civil infrastructure · Weakly supervised learning · Semantic segmentation · Model-assisted labeling

1 Introduction

Until recently visual inspection was exclusively a manual process conducted by reliability engineers. Not only is this dangerous due to the complexity of many civil engineering structures and the fact that some parts are hardly accessible. The main objective of the inspection is to assess the condition of an asset and determine whether repair or further maintenance operations are needed. Specifically, engineers make such decisions by analyzing the surfaces in search for defects, such as cracks, spalling, rust or algae, and assessing their severity, relative to their size and location in the structure.

K. Janouskova—Part of the work was done during the author's internship at IBM Research Zürich.

The advances in drone technology and its falling costs have recently pushed this laborious process of manual inspection progressively towards automation. Flying drones around a structure and using embedded high-resolution cameras to collect visual data from all angles not only speeds up the inspection process, but it also removes the human from potentially dangerous situations. In addition, thanks to the power of artificial intelligence capabilities, defects can be detected and localized with high precision automatically and presented to the reliability engineer for further analysis.

Typical approaches go beyond defect detection and generate fine-grained segmentation masks, which better characterize the defect. However, the drawback of these segmentation models is that they are fully supervised and therefore require a significant volume of high quality annotations at training time. Generating fine-grained segmentation masks is a manual task that involves a human expert deciding whether each pixel in the image belongs to a defect or not, which is time consuming and error-prone. Depending on the size of the images captured during inspection and the volume of defects present in a single image, annotating all defects per image can take hours, even with the aid of annotation tools like CVAT [26] or SuperAnnotate [30]. For example, it has been reported that single large (2048×1024) images depicting complex scenes require more than 90 min for pixel-level annotation [7].

The need for such expensive annotations can be alleviated by weakly supervised learning, in which a neural network is trained with cheaper annotations than explicit localization labels. In particular, weakly supervised segmentation methods can use image-level class labels [2,5,8,17], which require a single pixel annotation within the localized region of the target object. By using attribution maps obtained from a classifier, such as Grad-CAM [27], it is possible to identify the most important and discriminative regions of an image. However, these generated maps do not tend to cover the entire region of the target objects. Typical attempts to extend the maps manipulate either the image [19,29], or the feature map [11,32].

In this paper, we employ a different approach, based on adversarial-climbing, to extend the attributed regions of a target object [18]. This is opposed to an adversarial attack, which generates small perturbations of an image in order to change its classification output. As a result of applying adversarial climbing iteratively, the attribution map of the image gradually focuses on more extended regions of the target object, and can be used to generate fine-grained segmentation masks.

Specifically, we build a framework for model-assisted labeling of defects detected as a result of visual inspections of bridge structures from high resolution images. We train a classifier to recognize defect labels, apply Grad-CAM to generate segmentation masks and refine these masks with adversarial climbing. Once the masks have been generated, they are made available to the user through an interaction tool, where the expert is able to visualize, accept, reject or correct them, if need be (Fig. 1). We evaluate the approach on a real-world dataset and show that even after the first iteration, more than 50% of annotators' time is saved by refining the obtained masks instead of manually generating them. Moreover, the time saved is expected to increase in further iterations.

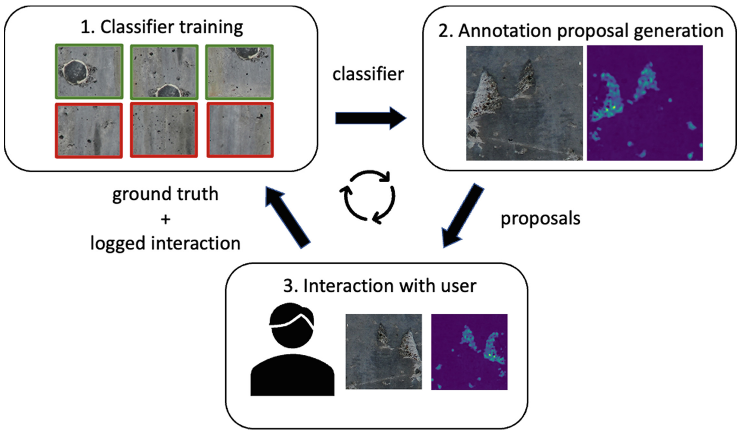

Fig. 1. Overview of the assisted labeling framework. First, a classifier is trained on weakly annotated images. Second, the classifier generates annotation proposals. Finally, the user interacts with the proposal (accept/modify/reject). The interaction is logged and used to extend the training set and improve the classifier, resulting in improved annotation proposals on future data.

2 Related Work

2.1 Weakly Supervised Segmentation and Localization

The vast majority of weakly supervised semantic segmentation and object localization methods depend on attribution maps obtained with approaches like Grad-CAM [27] from a trained classifier. While identifying the relevant regions of an image that have contributed to the classifier's decision is the goal, these regions tend to not be able to identify the whole region occupied by the target object.

Therefore, there have been many attempts to extend the attributed regions to ensure they cover more of the detected object. One popular approach is to manipulate the image [19,29]. For instance through erasure techniques [11,13, 22,31,32], already identified discriminative regions of the image are removed in an iterative manner, thus forcing the classifier to identify new regions of the object to be detected. However, the main drawback of the erasure approach is that there is a risk to generate wrong attribution maps when by erasing the discriminative regions of an image, the classifier's decision boundary changes. An alternative to image manipulation is feature map manipulation [6,16]. This produces a unified feature map by aggregating a variety of attribution maps from an image obtained by applying dropout to the network's feature maps.

Recently, adversarial climbing has been proposed to extend the attributed regions of the target object [18]. Applied iteratively to the attribution maps of a manipulated image results in a gradual identification of more relevant regions of the object. Regularization is additionally applied to avoid or reduce the activation of irrelevant regions, such as background or regions of other objects.

Unlike other approaches that require additional modules or different training techniques, applying adversarial climbing acts essentially as a post-processing step on top of the trained classifier. This makes it possible and easy to replace the underlying classifier's architecture or improve its performance without performing any changes to the backbone.

While adversarial climbing has been mainly applied for semantic segmentation, we employ it for instance segmentation, to generate precise and high-quality segmentation masks for fine-grained defects present in civil infrastructures. These masks go beyond providing localization cues for weakly supervised instance segmentation and defect localization. They significantly reduce the time required to manually annotate such defects at pixel-level, thus enabling downstream tasks such as supervised defect detection and segmentation at much lower costs.

2.2 Annotation Tools

Many annotation tools successfully deploy semi-supervised interactive annotation models, with different level of weak supervision at inference time. Traditional methods like GrabCut [25] which do not require fully-supervised pre-training exist, but are outperformed by learning-based strategies.

In DEXTR [23], at least four extreme points are required at inference time to infer segmentation while a bounding box and up to four correction points are used in [4]. In [14], a single click on an instance is enough to generate its segmentation mask. A crucial disadvantage of these approaches is that they do not have any localization ability and the detection of the defects fully relies on the human annotator. The performance of learning-based models also typically decreases with domain transfer. To obtain good performance on a new domain, full annotations are required. In [1], the problem of domain transfer is tackled by online fine-tuning.

3 Model-Assisted Labeling Framework

Instead of requiring segmentation masks from annotators, we propose to use weak labels consisting of classification labels. One label per image would make GPU training extremely challenging due to the large size of images in our dataset. Therefore, we ask the annotators to localize patches that contain defects. This approach is still substantially faster to input since they only require one click per defect. Similarly, negative samples require one click to indicate the absence of defects within a given image patch. These inputs are then used to generate a training dataset for a defect classifier by sampling crops around the annotated pixels.

Common approaches to weakly supervised learning with class-level supervision [3] use encoder architectures such as ResNet [10] to generate class activation maps (CAMs) [27]. Some work has gone into improving the resolution of the obtained CAMs using multi-scale inference [3,18] followed by post-processing steps like dense CRF [15], or aggregating activations from different levels of a ConvNet [9,12,28]. In a semi-supervised setup, [20] adopt a U-net architecture [24] and pre-train the encoder on a classification task, and then train the decoder to improve the mask starting with CAMs as a segmentation prior.

3.1 Proposal Generation and Refinement

Our weakly-supervised method to generate fine-grained segmentation masks consists of two steps. First, a deep neural network trained on a classification task is used to generate CAMs. Second, the CAMs go through a simple post-processing step to remove noise before connected component analysis, which gives the final annotation proposals. We generate the CAMs for all images as rejecting false positives only takes a negligible amount of time compared to polygon annotation and it forces the human annotators to check all the images for false positives/negatives. Optionally, most false negatives could easily be filtered out by applying a classifier to image patches. An example of the initial CAMs and the post-processed output is shown in Fig. 2.

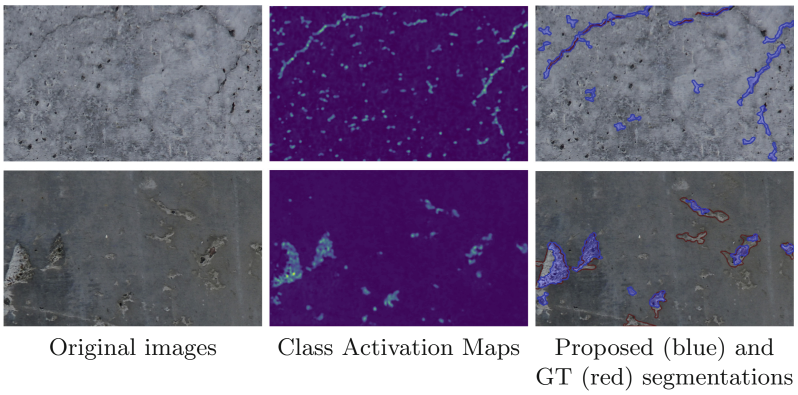

Fig. 2. Example images for the class ‘crack’. First column: two examples of image of concrete surfaces containing cracks, and therefore labeled with the classification label ‘crack’. Middle column: corresponding CAMs. Last column: corresponding proposed annotations obtained by filtering out noise in the CAMs by post-processing (blue), Ground Truth (GT) segmentation masks obtained by expert manual annotation from scratch (red). The proposed annotations generated by our method have large overlap with GT segmentation masks provided by expert annotators. (Color figure online)

Model Architecture. Similarly to [20], we adopt U-net [24], a segmentation architecture which aggregates features from different levels of the encoder at different resolutions. Instead of only using the pre-trained encoder for classification, we add the classification head directly on top of the decoder. This approach brings the advantage of having weights pre-trained on the target data as an initialization for subsequent fully supervised learning and increases the resolution of the final layer CAMs.

To further improve the resolution of the CAMs, we only build the U-net on top of the first two blocks of a Resnet34 encoder, avoiding resolution degradation from further downsampling. We also set the stride to 1 instead of the original 2 in the first convolutional layer, before the residual blocks. The model reduction does not lead to any classification performance degradation for the target application, however, Resnet34 produced better quality attribution masks then Resnet18. An overview of the architecture can be found in Fig. 3 and examples of CAMs with a different number of downsampling layers in Fig. 4.

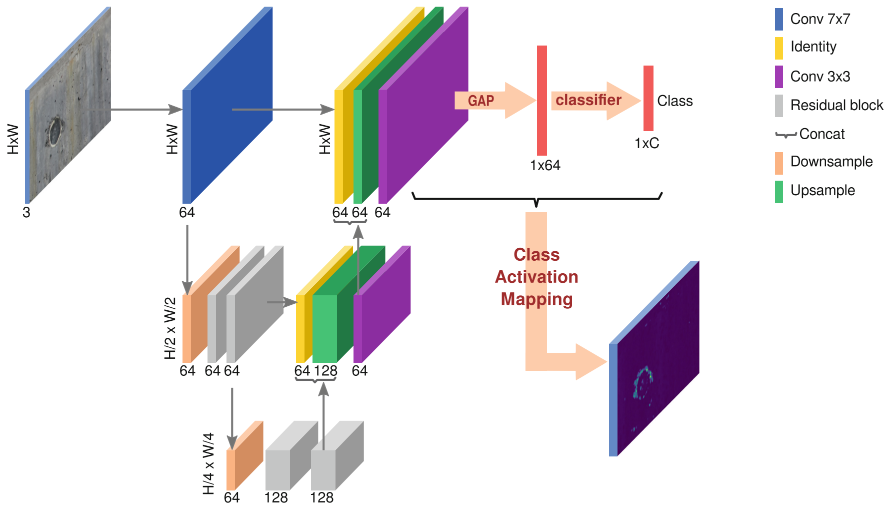

Fig. 3. Our U-net-based classifier for assisted labeling via explainability. Using a U-net architecture as the feature extractor of the classifier trained on (weak) classification labels allows us to generate CAMs with the same resolution as the input images using Grad-CAM, a standard gradient-based explainability method. The obtained high-resolution CAMs are then refined using anti-adversarial climbing (AdvCAM), post-processed, and used as proposal segmentation masks that can be further refined by annotators in a standard annotation tool.

Masks as Attribution Maps. GradCAM [27] is a gradient-based attribution method used to explain the predictions of a deep neural classifier by localizing class-discriminative regions of an image. It can be used for any layer but in the context of weakly supervised segmentation, the attribution maps of the final layer are typically used. When the last convolutional layer is followed by global average pooling (GAP) and a linear layer as classifier, these maps are commonly referred to as class activation maps (CAMs).

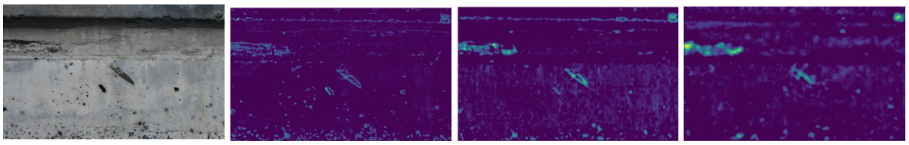

Fig. 4. Sparsity vs. CAM resolution trade-off. 'Spalling' CAMs of networks with one, two and three downsampling layers are shown. The less downsampling layers, the sparser and with better resolution the maps are.

Refinement with Adversarial Climbing. The attribution maps obtained with GradCAM typically only reflect the most discriminative regions of an object. Recent methods aim to mitigate this in different ways, by extending the attributed regions of the target object. In our framework, we adopt AdvCAM [18], which produces the CAMs on top of images obtained by iterative anti-adversarial manipulation, maximizing the predicted score of a given class while regularizing already salient regions. We observe that for the civil infrastructure domain, a much smaller number of iterations (2) is needed as opposed to the original work (27) to output CAMs with the entire (or almost entire) object region covered. Ideally, as the optimal number of iterations may vary per image, the number of iterations is adjustable by the user in an annotation tool.

The idea of AdvCAM is inspired by that of an non-targeted gradient adversarial attack where a small perturbation is applied to an image x so that the perturbed image x' confuses the classifier into predicting a different class:

$$x' = x - \xi \nabla_x \mathrm{NN}(x). \tag{1}$$

In AdvCAM, instead of minimizing the score of the target class c, the goal is to maximize it by applying:

$$x' = x + \xi \nabla_x y_c \tag{2}$$

where y_c is the logit of the target class.

This is referred to as anti-adversarial climbing and the procedure is iterative. Two forms of regularization are also introduced: i) the logits of the other classes are restricted to avoid increase in score for objects of classes close to the target class, and ii) attributions of already salient regions are restricted so that new regions are discovered.

Finally, the CAMs obtained from the adversarially-manipulated images are summed over all iterations and normalized. For more details, please refer to AdvCAM [18].

Post-processing. However, after the previous refinement step, the resulting CAMs contain noise, especially for images with highly structured background. Adversarial climbing typically further increases the amount of noise. Single threshold binarization either includes the noise for lower values, or defect parts are suppressed alongside the noise for higher values. Due to the increased resolution, the resulting activation maps are also sparser, sometimes leading to a defect split into multiple parts, especially for very thin cracks. We add two fast and simple post-processing steps after binarization with a low threshold value, $\theta = 0.1$, to address these issues. First, morphological closure is applied to counter the sparsity. Second, connected components are retrieved from the mask and all regions with an area below a threshold are filtered out. These steps effectively remove the majority of the noise while retaining the defect regions. The threshold value θ was selected according to the best performance on the validation set. The closure filter and minimal component area were selected based on observation of qualitative results. An example illustrating these steps and the post-processed result is shown in Fig. 5.

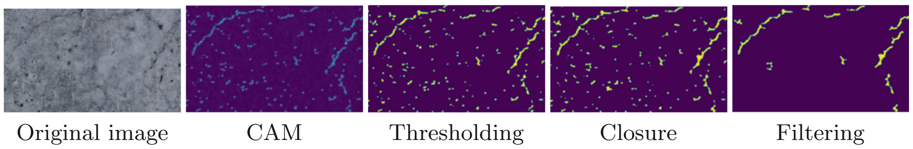

Fig. 5. Postprocessing for noise removal from CAMs. The figure shows the essential post-processing step from the initial CAMs to the final proposed annotation. The CAMs are first binarized with a low threshold (in this case $\theta = 0.1$). We then apply morphological closure to the binary map, followed by filtering of connected components by area to remove small clusters of pixels.

3.2 Interactive Aspects

The quality of the automatically generated annotations varies considerably. We therefore treat them as proposals to be screened by human annotators in a selection phase, before training a segmentation model on them. The generated annotations are split into three groups:

- *Accept* – the proposed annotation covers the predicted defect and no modifications to the defect mask are needed;
- *Modify and accept* – the proposed annotation covers the predicted defect, but the mask needs refinement;
- *Reject* – the proposed annotation does not contain the predicted defect, as it is a false positive. Furthermore, the annotator checks if some defects have been missed by the model (false negatives), in which case a manual annotation process should take place.

Annotation modification. Depending on the proposed defect mask and the capabilities of the annotation tool used, the modification can take multiple forms. The most common one is the removal of false positive patches of "noise" from the neighbourhood of the defect, which can be done by simply clicking on them to erase them. Another commonly needed modification is removing or adding a

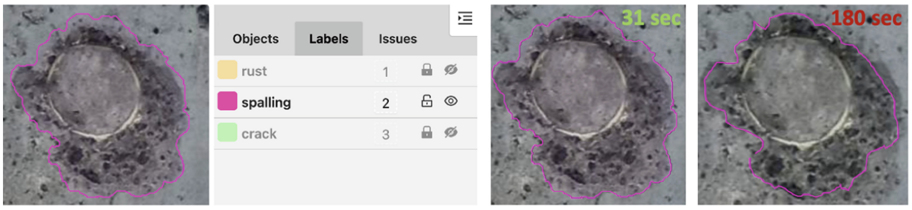

Fig. 6. Difference in labeling time between manually refining a proposed annotation obtain with our method vs. manual annotation from scratch. The proposed annotation obtained through our method for a spalling defect is loaded in CVAT (left). It is manually refined by hand by a human annotator in 31 s (center). On the other hand, manual annotation from scratch of the same defect takes ~6 times longer (180 s) than refining the proposed annotation (right).

small part of the defect, which requires moving/adding/removing polygon points, or with a brush tool. The interaction time can be further reduced by more sophisticated operations, for example, merging and splitting components, mask erosion and dilation, if the annotation tool in use allows for such operations.

Interaction logging. The interaction of the annotator with the proposals provides valuable information that can be exploited if logged. For instance, flagged false positive and false negative regions could be used to extend the weakly supervised training set, which consequently would improve the classifier and the subsequent generated proposals. The time spent on the modification of a proposal until it is accepted can also be used as a proxy for the sample difficulty, allowing for more efficient training strategies. This working modality of our proposal could be used in the future as part of a labeling pipeline combining active learning pipeline and weak supervision.

Demonstration. We show user interaction in CVAT, an open source annotation tool extensively used in various domains, including civil infrastructure. There, experts are able to visualize, accept, modify and reject the proposed annotations resulting from our framework. In Fig. 6, refining an annotation proposal for a spalling defect takes 31s, as opposed to manually generating the mask, which takes 180s. Additionally, the manual annotation is less fine-grained (i.e., fewer polygon points) than the proposal and reaching the same level of detail manually would extend well beyond 180s. The example shown here is pessimistic as the proposed and refined masks are very close and it is not clear edits were necessary. However, the refinement took only 16 % of the full annotation time.

4 Evaluation

4.1 Data Preparation

The defect dataset consists of 732 high resolution (5K $\times$ 5K pixels) images. The classification dataset is generated by sampling 5 crops of size 320 $\times$ 320 pixels around each positive (corresponding to a defect) user-annotated pixel. Given the extreme class-imbalance, where the least number of instances per class is 675 and the largest number of instances is 21,787 (see Table 1), we create a separate binary classification (defect/no defect) dataset for each defect type. Since there is significantly more negative pixel annotations, crops of negative class are sampled uniformly from each annotation so that there is the same number of positive and negative samples in each of the datasets. Performance of classifiers trained on multiclass datasets created by over/undersampling was inferior.

Table 1. Number of instances of each class in the dataset (highly imbalanced).

Defect	Crack	Spalling	Rust
Instances	21,787	675	1,078

4.2 Classifier Training

The models share the same architecture and were implemented in the PyTorch framework. Each model was trained for 6 h on 2 Nvidia A100 GPUs with the batch size of 32 and the AdamW [21] optimizer (learning rate 1e-4, weight decay 1e-2, all other parameters were kept at PyTorch default values). The best checkpoint was selected according to the highest f-measure on the validation set.

4.3 Estimation of Time Saved

To quickly estimate the annotation time reduction due to our weakly supervised model, we used the following procedure where users only estimate the percentage of time saved, as opposed to actually annotating the data. We first assumed that the user has at their disposal standard annotation tools such as brush and erasure, as well as the possibility to apply morphological operations such as dilation and erosion. We then split the test set instances (i.e. the connected components of the segmentation ground truth masks in the test patches, which are 265 in total) into 3 groups according to the estimated percentage of the time saved annotating the instances when using the output of our method as an initial annotation proposal within such a standard annotation tool. Specifically, group G_{95} are the instances for which, by visual inspection, we estimated modifying the CAM via the annotation tool provided a time saving above 95% over annotating the instance from scratch. Analogously, groups G_{75} and G_{50} are groups of instances where we estimated a time saving above 75% (but below 95%) and above 50% (but below 75%), respectively. Examples of instances from each group for all defects are shown in Fig. 7.

The ratio between the total time saved and the time needed to annotate was then simply estimated from the number of instances in each of the groups of connected components as:

$$\text{Relative time saving} = \frac{1}{N} \sum_{i \in \{95,75,50\}} \frac{i}{100} |\mathrm{G}_i|, \tag{3}$$

where $|\mathrm{G}_i|$ is the number of instances in the group.

This formula allowed us to estimate an average reduction of 52% in annotation time. This is broken down as follows for different types of defects: 57% for cracks, 58% for spalling and 40% for rust. Detailed results of this estimation procedure are reported in Table 2. The relatively low time saving on rust can be explained by the high sensitivity of the annotation proposal shape to the value of the threshold applied to the CAMs, meaning that each component would require fine manual tuning of the threshold within the annotation tool.

An important note on the limitations of this time estimation procedure is that on one hand, it does not take into account the instances that were missed by our method, meaning that in practice we assumed recall of the defects. On the other, the time estimation is very conservative. Less than 50 % time reduction is considered as 0 and a lower bound is used for the rest of the intervals. The final result of 52% annotation time saved should thus be considered as a very conservative lower bound and we plan to conduct a more detailed evaluation in the future.

Table 2. Estimated annotation time saved by our method in percentage over annotating the defect instances from scratch on a test set of 265 instances. Relative time saving is shown for each type of defect separately and in total averaged over all defects. The number of instances where the percentage of time saved $t \in [95, 100]$, $t \in [75, 95)$ and $t \in [50, 75)$ is also reported. Anything less than 50 is considered as no time saved on the instance.

	Instance count	95	75	50	Time saved (%)
Crack	111	19	40	30	57
Spalling	65	17	19	14	58
Rust	89	22	14	9	40
All defects	265	58	73	53	51

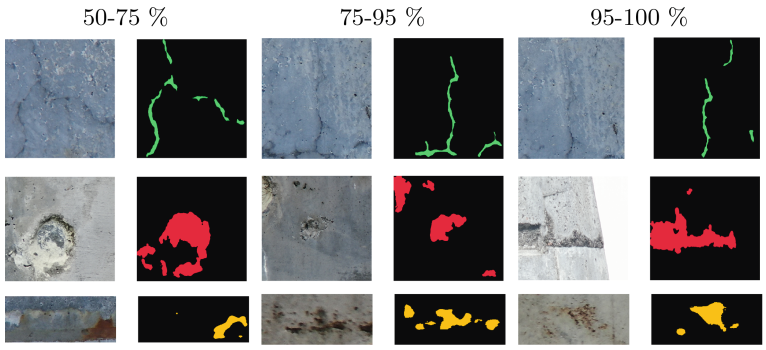

Fig. 7. Examples of instances according to the annotation time saved (in %). The first, second and last row show instances of cracks, spalling and rust, respectively.

5 Conclusions

We explored the use of weak labels from human annotators as a means to reduce the labeling time for a segmentation task in visual inspection, an application domain where the time of specialized annotators is particularly costly. The advantage of weak labels is that they are cheap to obtain because they require minimal interaction with the annotator. In our proposed approach, weak labels are used to train a classifier in order to generate proposals for segmentation masks by means of an explainability attribution method, followed by iterative adversarial climbing. Domain experts can then correct the proposed masks where needed by integrating this workflow in standard annotation tools like CVAT. Moreover, proposal segmentation masks can be used as pseudo-labels for unlabeled images, which can subsequently be employed to train supervised segmentation models, as well as to diagnose issues with ground truth labels from previous annotation campaigns.

Acknowledgement. This research was supported by Czech Technical University student grant SGS20/171/OHK3/3T/13. We would like to thank Finn Bormlund and Svend Gjerding from Sund&Bælt for the collection and annotation of the image data.

References

1. Acuna, D., Ling, H., Kar, A., Fidler, S.: Efficient interactive annotation of segmentation datasets with polygon-rnn++. In: CVPR, pp. 859–868 (2018)
2. Ahn, J., Kwak, S.: Learning pixel-level semantic affinity with image level supervision for weakly supervised semantic segmentation. In: CVPR (2018)
3. Ahn, J., Cho, S., Kwak, S.: Weakly supervised learning of instance segmentation with inter-pixel relations. In: CVPR, pp. 2209–2218 (2019)
4. Benenson, R., Popov, S., Ferrari, V.: Large-scale interactive object segmentation with human annotators. In: CVPR, pp. 11700–11709 (2019)

5. Chang, Y.T., Wang, Q., Hung, W.C., Piramuthu, R., Tsai, Y.H., Yang, M.H.: Weakly-supervised semantic segmentation via sub-category exploration. In: CVPR (2020)
6. Choe, J., Lee, S., Shim, H.: Attention-based dropout layer for weakly supervised single object localization and semantic segmentation. In: TPAMI (2020)
7. Cordts, M., et al.: The cityscapes dataset for semantic urban scene understanding. In: CVPR (2016)
8. E., K., Kim, S., Lee, J., Kim, H., Yoon, S.: Bridging the gap between classification and localization for weakly supervised object localization. In: CVPR (2022)
9. Englebert, A., Cornu, O., De Vleeschouwer, C.: Poly-cam: high resolution class activation map for convolutional neural networks. In: ICPR (2022)
10. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016)
11. Hou, Q., Jiang, P., Wei, Y., Cheng, M.M.: Self-erasing network for integral object attention. In: NeurIPS (2018)
12. Jiang, P.T., Zhang, C.B., Hou, Q., Cheng, M.M., Wei, Y.: Layercam: exploring hierarchical class activation maps for localization. IEEE Trans. Image Process. **30**, 5875–5888 (2021)
13. Ki, M., Uh, Y., Lee, W., Byun, H.: In-sample contrastive learning and consistent attention for weakly supervised object localization. In: ACCV (2020)
14. Koohbanani, N.A., Jahanifar, M., Tajadin, N.Z., Rajpoot, N.: NuClick: a deep learning framework for interactive segmentation of microscopic images. Med. Image Anal. **65**, 101771 (2020)
15. Krähenbühl, P., Koltun, V.: Efficient inference in fully connected crfs with gaussian edge potentials. NeurIPS **24**, 109–117 (2011)
16. Lee, J., Kim, E., Lee, S., Lee, J., Yoon, S.: Ficklenet: weakly and semi-supervised semantic image segmentation using stochastic inference. In: CVPR (2019)
17. Lee, J., Oh, S.J., Yun, S., Choe, J., Kim, E., Yoon, S.: Weakly supervised semantic segmentation using out-of-distribution data. In: CVPR (2022)
18. Lee, J., Kim, E., Yoon, S.: Anti-adversarially manipulated attributions for weakly and semi-supervised semantic segmentation. In: CVPR, pp. 4071–4080 (2021)
19. Li, K., Wu, Z., Peng, K.C., Ernst, J., Fu, Y.: Tell me where to look: Guided attention inference network. In: CVPR (2018)
20. Lin, D., Li, Y., Prasad, S., Nwe, T.L., Dong, S., Oo, Z.M.: CAM-UNET: class activation MAP guided UNET with feedback refinement for defect segmentation. In: ICIP, pp. 2131–2135 (2020)
21. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. In: ICLR (2019)
22. Mai, J., Yang, M., Luo, W.: Erasing integrated learning: a simple yet effective approach for weakly supervised object localization. In: CVPR (2020)
23. Maninis, K.K., Caelles, S., Pont-Tuset, J., Van Gool, L.: Deep extreme cut: from extreme points to object segmentation. In: CVPR, pp. 616–625 (2018)
24. Ronneberger, O., Fischer, P., Brox, T.: U-Net: convolutional networks for biomedical image segmentation. In: Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F. (eds.) MICCAI 2015. LNCS, vol. 9351, pp. 234–241. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24574-4_28
25. Rother, C., Kolmogorov, V., Blake, A.: "GrabCut": interactive foreground extraction using iterated graph cuts. ACM Trans. Graph. (TOG) **23**(3), 309–314 (2004)
26. Sekachev, B., et al.: opencv/cvat: v1.1.0 (2020). https://doi.org/10.5281/zenodo.4009388

27. Selvaraju, R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D.: Grad-cam: visual explanations from deep networks via gradient-based localization. In: ICCV, pp. 618–626 (2017)
28. Shinde, S., Chougule, T., Saini, J., Ingalhalikar, M.: HR-CAM: precise localization of pathology using multi-level learning in CNNs. In: Shen, D., et al. (eds.) MICCAI 2019. LNCS, vol. 11767, pp. 298–306. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-32251-9_33
29. Singh, K.K., Lee, Y.J.: Hide-and-seek: Forcing a network to be meticulous for weakly-supervised object and action localization. In: ICCV (2017)
30. SuperAnnotate: https://www.superannotate.com (2018)
31. Wei, Y., Feng, J., Liang, X., Cheng, M.M., Zhao, Y., Yan, S.: Object region mining with adversarial erasing: a simple classification to semantic segmentation approach. In: CVPR (2017)
32. Zhang, X., Wei, Y., Feng, J., Yang, Y., Huang, T.S.: Adversarial complementary learning for weakly supervised object localization. In: CVPR (2018)