Semantic segmentation with small training datasets: A case study for corrosion detection on the surface of industrial objects

Dennis Haitz, Patrick Hübner, Markus Ulrich, Steven Landgraf, and Boris Jutzi

Karlsruhe Institute of Technology (KIT)
Institute for Photogrammetry and Remote Sensing (IPF)
Englerstr. 7, 76131 Karlsruhe

Abstract In this research, we investigate possibilities to train convolutional neural networks with a small dataset for semantic segmentation, while achieving the best possible model generalization. In particular, we want to segment corrosion on the surface of industrial objects. In order to achieve model generalization, we utilize a selection of established and advanced strategies, i.e. Self-Supervised-Learning. Besides radiometric- and geometric-based data augmentation, we focus on model complexity regarding encoder and decoder, as well as optimal pre-training. Finally, we evaluate the best performing model against a pixel-wise random forest classification. As a result, we achieve an f1-score of 0.79 for the best performing model regarding the segmentation of corrosion.

Keywords Semantic segmentation, classification, machine vision, surface inspection, corrosion detection, quality assurance

1 Introduction

In the field of machine vision (MV), image segmentation techniques are heavily utilized for the surface inspection of industrial objects [1]. Image segmentation leads to image regions that can represent image texture in a geometrically precise manner. Well established segmentation methods like thresholding, clustering or region growing, however, have the disadvantage of lacking semantic information. Newer

deep-learning-based (DL) segmentation methods based on convolutional neural networks (CNN) are capable of adding semantics implicitly within the training process. These methods are often based on fully-convolutional-networks (FCN), which only consist of convolution layers as learnable layers, besides optional batch-normalization. FCNs can be viewed as functions that map an input image to a map of $n \in C$ scores per pixel, where C denotes a set of class labels. By applying an *argmax* function, the most likely class c is chosen for a particular pixel. While DL-based models outperform pre-DL methods on large datasets, the downside of such models is the potential of overfitting due to the large amount of model parameters. In a lot of practical applications, however, no adequate amount of data is available [2]. Among other applications, common MV tasks in the area of surface inspection lack a sufficient amount of data in order to train a DL-based model to generalize well. Recent advances in DL research target the challenges of small training datasets.
This work aims to utilize a selection of these advanced learning strategies as well as established methods in order to approximate the best possible model generalization. Our scenario includes a barrel as it is used for the storage of low radioactive waste (Figure 1(a)), which we from now on refer to as our object. The training set consists of an RGB image I_{Train} of the unwrapped coat of the object (Figure 1(b)), whereas the test set consists of an RGB image I_{Test} of the bottom. Both sets are labeled to separate the image pixels into eight classes. In our previous work [3], we already utilized the coat for training and also testing, though both datasets were from different areas of the coat and therefore disjunct from each other. In this new work, however, I_{Test} is aquired under different illumination conditions, which sets both I_{Train} and I_{Test} even further apart from each other regarding the image characteristic. For our scenario, we exclusively use I_{Train} and no additional image datasets or unlabeled data for training. Merely, we use I_{Train} without labels within a model training at some point in this work. To train a model, I_{Train} is split up into smaller image patches for model input. We employ established and widely used **data augmentation** (Section 3.1) techniques by applying geometric and radiometric image transformations. Another aspect of our work is **encoder pretraining** (Section 3.3). For this purpose, we train the models with randomly initialized model parameters according to some normal distribution and

with ImageNet-pretrained parameters. As a third encoder pretraining strategy, we employ self-supervised-learning (SSL) [4]. For measuring the impact of model complexity, we undertake a **model selection** (Section 3.2). Therefore, we use two encoders with different depth of the same model family: ResNet18 and ResNet50 [5]. The accompanying decoder architectures are U-net [6] and DeepLabv3 [7]. The stated techniques are stages in a training pipeline, where the best performing technique per stage gets chosen. For comparability, we also employ a pre-DL algorithm to evaluate the results of both learning domains against. A **random forest classifier** [8] (RF) (Section 3.4) therefore is applied within the RGB feature space. The result is a pixel-wise classification without further contextual information.

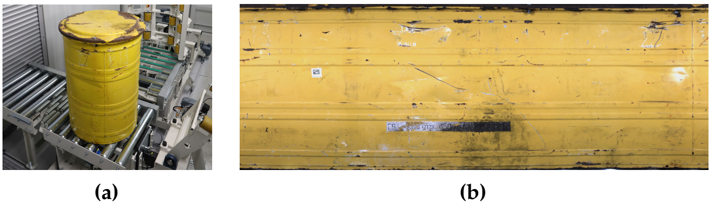

Figure 1: Barrel in the test facility (a). Within the facility, the image data of the coat and bottom is acquired. Image of the unwrapped coat (b), used for training our models.

2 Related work

The automated detection of structural damage such as corrosion on industrial objects based on image data is an active field of research [9]. Specifically, many research efforts are focused on applying end-to-end DL to the task of corrosion detection [10]. This, however, poses the challenge that DL approaches are typically data-hungry, requiring large amounts of training data, while publicly available, labeled datasets for corrosion detection are few and far between [11]. Furthermore, the visual appearance of corrosion is quite specific w.r.t. the respective target materials and shapes and it is still an open research question to what extent the recently published dataset from [12] can be transferred to specific application scenarios such as the coated steel barrels used in our work. Thus, while other research addresses the question of how to alleviate the effort of creating large-scale training datasets for corrosion detection, e.g. via crowd-sourcing [13] or efficient labeling tools [14], we focus on how to efficiently use small amounts of training data in a DL context by evaluating the impact of pretraining methods from the fields of SSL in relation to the results of state-of-the-art DL networks on a common small-scale dataset.

DL-based corrosion detection can be approached as a classification problem, where image regions are classified w.r.t. the presence of corrosion in a sliding window manner [15] . Sometimes, the results of a sliding window classification are further post-processed to yield pixel-wise segmentation results, e.g. via the activation maps of patches that have been classified as containing corrosion [16]. Other works aim at detecting corrosion by means of DL-based object detection networks such as R-CNNs [17]. Here, first, instance-wise bounding boxes are regressed which are subsequently refined to pixel-wise segmentation masks. Lastly, as is the case in our work, DL-based corrosion detection can be approached as a semantic segmentation task. In [18], different fully convolutional segmentation networks are comparatively evaluated for the task of segmenting corrosion spots on steel structures. In [19], fully convolutional segmentation is compared against an approach based on R-CNN. As the results are found to not be precise enough, they are refined by a contour-aware postprocessing approach. Lastly, [20] apply DeepLabv3 in a multi-temporal setting for damage-progress monitoring.

3 Methodology

In this section, the methods and the utilized datasets are described. Two methodological strings are applied: One string represents the model pretraining with the application of all possible encoder-decoder-combinations. Aside from that, these models are trained with the baseline dataset, as well as the augmented dataset. The last string is a pixel-wise RF classification within the RGB feature space.

3.1 Data augmentation

As mentioned previously, the baseline dataset for training consists of 102 image patches of size 512×512 px. In the data augmentation process, geometric and radiometric transformations are applied to these patches. The geometric transformations consist of rotations, as well as a combined crop and resize operation. Because CNNs are rotation invariant to only some degree, the distinction between an image patch and its rotated variant should have a positive effect on the generalization capability. The second geometric transformation is a combined crop and resize operation. A crop of an image is chosen randomly and then resized to the original image patch size. The resizing operation utilizes a bilinear interpolation. With this combination, we aim at creating new appearences of texture, which differ from the original image patch. Finally, the radiometric transformation consists of a color space transformation to HSI, where saturation and intensity are randomly varied. The image patch then gets transformed back to RGB. This strategy is applied to simulate different illumination situations.

3.2 Model selection

The model complexity is one aspect of our investigation. Usually in ML, in order to prevent overfitting, one strategy is to reduce the model complexity, or to be more specific, the number of model parameters. In the case of DL-based models, one possibility to achieve this is to consider different depths of a model. Another aspect is the selection of a decoder, which is responsible for upsampling the learned features to a map of classification scores with the size of the original image.
Encoders. We utilize the ResNet architecture [5] for our investigations. This architecture is found quite often in literature as a standard model. ResNets are used with different depths. We employ a ResNet18 as the *small* encoder with a rather low complexity. The ResNet50 on the other hand is selected as the *large* encoder, as it contains 50 convolution layers. Large encoders have the advantage of learning more distinct features in the lower convolution layers, but have more parameters to optimize as a disadvantage regarding small training sets. Because the surface textures of our object are not very complex, we aim for better generalization while not requiring such distinct features by applying

the ResNet18 encoder.
Decoders. For similar reasons as mentioned before, we select the U-net and DeepLabv3 architectures as the decoders, as these are commonly used in the domain of semantic segmentation. The U-net has a feature preservation aspect to it, because of the so-called skip-connections. These skip-connections map the output of a convolution layer to its corresponding transpose-convolution layer on the decoding side. The DeepLabv3 achitecture applies so-called atrous convolutions and atrous spatial pyramid pooling. The former is applied to yield a more dense feature representation in the upscaling process. The latter is applied to include scale invariance to some extent.

3.3 Encoder pretraining

To pretrain the encoder, we apply three different methods: random initialization according to a normal distribution, pretraining on the ImageNet dataset and SSL. For the latter method, this is achieved by training an encoder model within an SSL model and then by applying transfer learning, in order to embed the pretrained encoder into the segmentation architecture, which is done by extracting the encoder from the SSL model and append a decoder afterwards.
The **random initialization** often is the default in popular frameworks in contrast to setting the parameters to some constant value. In our case, the parameters are initialized according to the normal distribution parameterization described in [5], with $\mathcal{N}(0, \frac{2}{n_l})$, where n is derived from the number of input features as well as the filter size and l as a layer index.
ImageNet pretraining is popular, because of the transfer learning aspect. Only the features on the first layers of training are of interest because there, low-level features like edges, point-like shapes or corners are already learned. This can help for faster convergence or maybe even convergence at all. Of course, pretraining on other datasets is also possible, especially if they are semantically related to the follow-up training domain.
The field of **self-supervised-learning** is densely connected to the DL-field with a highly active ongoing research. An SSL model is trained on exclusively unlabeled data. In our case, a contrastive SSL method is applied: SimCLR [21]. This method takes a sample out of the dataset

as a positive sample, and another disjunct sample as the negative sample. For higher distinctiveness, also data augmentation strategies are applied to the positive sample. A contrastive loss is then calculated from both samples for backpropagation. The purpose of SSL methods is to learn feature representations without knowledge about semantics. This is called the pretext task. From there, the underlying model can be extracted and added to a so-called downstream task. This procedure can be viewed as a transfer learning. The downstream task in our case is the semantic segmentation, where the SSL pretrained encoder is embedded into.

3.4 Random forest classification

The application of an RF classifier in the RGB feature space is done for evaluation. DL methods outperform pre-DL learning techniques on benchmark datasets in the most cases. In our use-case with a comparatively low amount of data, however, such methods might still outperform DL models.

4 Experiments

This section describes our experimental setup in the domain of methodology. Our goal is to detect corrosion as segmented image regions. The other classes are rejection classes and therefore not of further interest. We have four classes in total: lacquer, dirt, spots and corrosion. In our investigations we found that an over-classification leads to a better separability between corrosion and non-corrosion.
As mentioned in the Section 1, we use the image of the unwrapped barrel coat I_{Train} as our training dataset. It is an RGB image of size 3072×8763 px. For the specification of image size we use the notation of $height \times width$ throughout this work. The models are trained with smaller image patches with no overlap to neighboring patches, cut from I_{Train}. Those image patches are of size 512×512 px. With this size, we want to preserve as much information of the surface texture as possible through keeping spatial coherence. Splitting I_{Train} into patches results in 102 image patches as a baseline training dataset, which is used for training in the setting of no **data augmentation**. By applying data aug-

mentation, the dataset grows to 8160 image patches.
For the first variant of **encoder pretraining**, the encoder is not pretrained, but initialized randomly. We keep the default procedure of PyTorch, which distributes the model parameters according to the so-called Kaiming initialization [5]. For ImageNet-based encoder pretraining, we download the pretrained model parameters from torchvision. The SSL pretraining is done using a batch-size of 512 for both encoders. The dataset in both cases is the unlabeled baseline dataset. It is trained for 8000 epochs. It should be noted that both the ResNet 18 and ResNet50 are randomly initialized for the SSL pretraining.
The **DL model training** is organized using different combinations with and without data augmentation, with ResNet18 and ResNet50 and with three different pretraining settings for the encoder, namely randomly initialized, ImageNet-, and SSL pretrained. At last, the number of the previously mentioned combinations is doubled be employing a U-net and DeepLabv3 decoder architecture. In sum, 24 models are trained and evaluated.
For **random forest training**, we applied 100 trees with a maximum depth of 8. The dataset used for this training is the baseline dataset with no augmentations. The reason is that the number of datapoints (pixels) is sufficient for pre-DL models to generalize well on the one hand, but also on the other hand, RFs are fairly robust against overfitting in general.
The **evaluation** uses the classification metrics precision, recall, f1-score and overall accuracy. These metrics show the performance for pixel-wise classification for each of the methods in order to make them comparable. Another metric for measuring the global per-class overlap of correct classified regions is the Intersection Over Union (IoU). Further on, the mean IoU (mIoU) as another metric is calculated by averaging all class-specific IoU values.

5 Results

In this section the results of our experiments are shown. Qualitative results in the form of visualizations are depicted in Figure 2(a) to 2(f). For quantitative results Table 1(a) shows the global metrics of all trained models. Table 1(b) shows the best performing DL model and the RF model with class-wise metrics each.

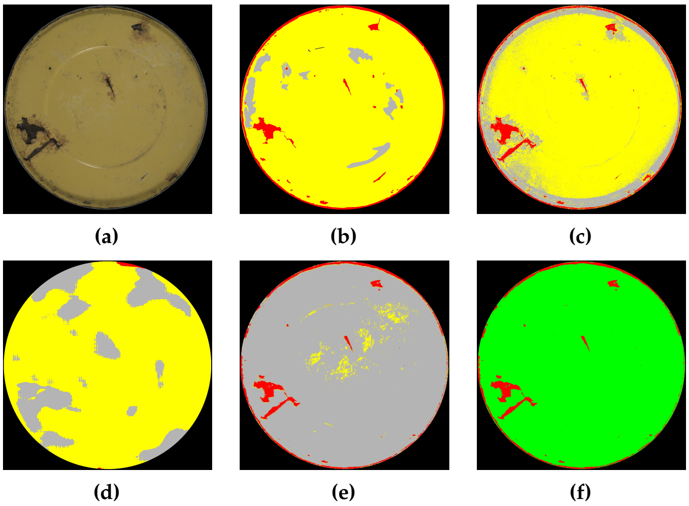

Figure 2: I_{Test} (a), ground truth (b), result of the RF classification (c). Ground truth and predictions are colored as: lacquer (yellow), dirt (bright gray), spots (dark gray), corrosion (red). Predictions of the DL models: worst performing model (*rn50-dl-noaug-inet*) (d), best performing model with all classes (*rn50-u-noaug-rand*) (e), best performing model with two aggregated classes no corrosion (green) and corrosion (red).

6 Discussion

Regarding the **model complexity**, ResNet18 and ResNet50 yield comparable results for the global metrics. For the detection of corrosion, however, ResNet50 usually shows better results. This holds true for the four best performing models concerning the f1-score of the corrosion class. This indicates that lower model complexity does not necessarily lead to better model generalization as proposed in Section 3.
DeepLabv3 and U-net decoders seem to be on par regarding global metrics, as well as for the corrosion detection. The highest f1-score for the corrosion class is achieved by a U-net model. Further on, DeepLabv3 seem to yield more smoothed results in the visual domain,

Table 1: The **global metrics** (a) of each model are shown. The model names are encoded as follows: *encoder-decoder-augmentation-pretraining*. The metrics are mean f1-Score (mF1), overall accuracy (OA) and mean Intersection over Union (mIoU). The marked model is not the best performing regarding the global metrics, but the best performing for the corrosion class. The **class-wise metrics** (b) are shown for the best performing DL model regarding the f1-score of the corrosion class, and the RF model. In addition to f1-score (F1) and intersection over union (IoU), precision (P) and recall (R) are depicted.

(a)

Model	OA	mF1	mIoU
rn18-u-noaug-rand	0.52	0.29	0.35
rn18-u-noaug-inet	0.65	0.37	0.48
rn18-u-noaug-ssl	0.75	0.41	0.60
rn18-dl-noaug-rand	0.81	0.30	0.68
rn18-dl-noaug-inet	0.39	0.20	0.24
rn18-dl-noaug-ssl	0.63	0.27	0.46
rn18-u-aug-rand	0.86	0.36	0.76
rn18-u-aug-inet	0.91	0.39	0.83
rn18-u-aug-ssl	0.82	0.29	0.70
rn18-dl-aug-rand	0.88	0.37	0.79
rn18-dl-aug-inet	0.89	0.39	0.80
rn18-dl-aug-ssl	0.88	0.43	0.78
rn50-u-noaug-rand	**0.10**	**0.21**	**0.05**
rn50-u-noaug-inet	0.80	0.41	0.67
rn50-u-noaug-ssl	0.54	0.38	0.37
rn50-dl-noaug-rand	0.39	0.24	0.24
rn50-dl-noaug-inet	0.75	0.24	0.61
rn50-dl-noaug-ssl	0.63	0.27	0.46
rn50-u-aug-rand	0.86	0.36	0.76
rn50-u-aug-inet	0.91	0.39	0.83
rn50-u-aug-ssl	0.82	0.29	0.70
rn50-dl-aug-rand	0.92	0.42	0.85
rn50-dl-aug-inet	0.86	0.40	0.76
rn50-dl-aug-ssl	0.88	0.40	0.79
random forest	0.80	0.44	0.67

(b)

Model	Class	P	R	F1	IoU
rn50-u-noaug-rand	Laquer	0.86	0.03	0.05	0.10
	Dirt	0.05	0.99	0.09	0.04
	Spots	0.00	0.00	0.00	0.00
	Corrosion	0.83	0.63	0.71	0.21
random forest	Laquer	0.95	0.83	0.86	0.82
	Dirt	0.06	0.21	0.09	0.05
	Spots	0.00	0.00	0.00	0.00
	Corrosion	0.82	0.76	0.79	0.42
rn50-u-noaug-rand	No Corrosion	0.98	0.99	0.99	0.90
	Corrosion	0.83	0.63	0.71	0.21
random forest	No Corrosion	0.99	0.99	0.99	0.96
	Corrosion	0.82	0.76	0.79	0.42

whereas some U-net-based models tend to show slightly more scattered segmentation results.

A surprising insight is that **data augmentation** did not seem to have a positive effect for all models. Moreover, we could only observe in the three best models, regarding f1-score in the corrosion class, that DeepLabv3 decoders benefit from data augmentation and tend to perform poor without data augmentation, while this tends to be the opposite case with U-nets.

For the **encoder pretraining**, we could not observe tendencies re-

garding the different pretraining strategies resulting in a superior performance. This is especially of interest, because random initialization is usually considered as an inferior starting point for training. In our experiments, the random initialization performs similar w.r.t. the other pretrainings. In literature, usually thousands of unlabeled images are utilized for SSL. As can be seen in Table 1(a), no gain could be achieved with SSL pretraining. It can be assumed that the 102 image patches were too few for a substantial SSL pretraining.
The **random forest classification** yields the best results regarding the f1-score of the corrosion class. It needs to be considered, however, that the RF classifier does not take context into account in our experiments. This leads to results with less smoothness in some regions where the separability in RGB space is not very pronounced. Especially larger areas of corrosion are prone to false negatives in the form of scattered pixels belonging to other classes.

7 Conclusion

For our applied strategies in order to train a DL model to generalize from a small baseline dataset, we found that for the core class of corrosion, a RF classifier performs better within the RGB feature space than a DL-based model. The RGB feature space in our case is well separable: There is no surface texture with a similar radiometric signature to that of corrosion in I_{Test}. Also, for the incorporation of context in the non-DL domain, a conditional random field could be of advantage. For the enrichment of the feature space, textural features can be extracted and added for training.
For the DL domain we found that there is still a large potential for improvement. While strategies like data augmentation are mandatory for a long time in such scenarios, we could not see a significant advantage. We only touched the surface of what is possible, with mediocre results at this point. Other possibilities are to incorporate unlabeled datasets for Semi-Supervised-Learning or a large scale Self-Supervised-Learning for better encoder pretraining. Also, Few-Shot-Semantic-Segmentation techniques can be taken into account in the future, as there is a fairly high research activity in this area.

References

1. C. Steger, M. Ulrich, and C. Wiedemann, *Machine Vision Algorithms and Applications*, 2nd ed. Wiley-VCH Verlag, 2018.
2. M. Heizmann, A. Braun, M. Glitzner, M. Günther, G. Hasna, C. Klüver, J. Krooß, E. Marquardt, M. Overdick, and M. Ulrich, "Implementing machine learning: chances and challenges," *Automatisierungstechnik*, vol. 70, no. 1, pp. 90–101, 2022.
3. D. Haitz, B. Jutzi, P. Hübner, and M. Ulrich, "Corrosion Detection for Industrial Objects: From Multi-Sensor System to 5D Feature Space," *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. XLIII-B1-2022, pp. 143–150, 2022.
4. L. Jing and Y. Tian, "Self-Supervised Visual Feature Learning With Deep Neural Networks: A Survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 11, pp. 4037–4058, 2021.
5. K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.
6. O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, Eds. Cham: Springer International Publishing, 2015, pp. 234–241.
7. L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 4, pp. 834 – 848, 2017.
8. L. Breiman, "Random Forests," *Machine Learning*, vol. 45, pp. 5–32, 2001.
9. S. K. Ahuja and M. K. Shukla, "A Survey of Computer Vision Based Corrosion Detection Approaches," in *International Conference on Information and Communication Technology for Intelligent Systems*, 2017, pp. 55–63.
10. W. Nash, T. Drummond, and N. Birbilis, "A Review of Deep Learning in the Study of Materials Degradation," *npj Materials Degradation*, vol. 37, pp. 1–12, 2018.
11. E. Bianchi and M. Hebdon, "Visual structural inspection datasets," *Automation in Construction*, vol. 139, pp. 1–18, 2022.
12. B. Yin, N. Josselyn, T. Considine, J. Kelley, B. Rinderspacher, R. Jensen, J. Synder, Z. Zhang, and E. Rundensteiner, "Corrosion Image Data Set for

Automating Scientific Assessment of Materials," in *British Machine Vision Conference (BMVC)*, 2021, pp. 1–15.

13. W. T. Nash, C. J. Powell, T. Drummond, and N. Birbilis, "Automated Corrosion Detection Using Crowdsourced Training for Deep Learning," *Corrosion*, vol. 76, no. 2, pp. 135–141, 2020.
14. A. Rahman, Z. Y. Wu, and R. Kalfarisi, "Semantic Deep Learning Integrated with RGB Feature-Based Rule Optimization for Facility Surface Corrosion Detection and Evaluation," *Journal of Computing in Civil Engineering*, vol. 35, no. 6, pp. 04 021 018:1–15, 2021.
15. T. Papamarkou, H. Guy, B. Kroenck, J. Miller, P. Robinette, D. Schultz, J. Hinkle, L. Pullum, C. Schuman, J. Renshaw, and S. Chatzidakis, "Automated Detection of Corrosion in Used Nuclear Fuel Dry Storage Canisters Using Residual Neural Networks," *Nuclear Engineering and Technology*, vol. 53, no. 2, pp. 657–665, 2021.
16. B. Burton, W. T. Nash, and N. Birbilis, "RustSEG – Automated Segmentation of Corrosion Using Deep Learning," *arXiv*, pp. 1–28, 2205.05426.
17. S. K. Fondevik, A. Stahl, A. A. Transeth, and O. Øystein Knudsen, "Image Segmentation of Corrosion Damages in Industrial Inspections," in *IEEE 32nd International Conference on Tools with Artificial Intelligence (ICTAI)*, 2020, pp. 787–792.
18. L. D. Duy, N. T. Anh, N. T. Son, N. V. Tung, N. B. Duong, and M. H. R. Khan, "Deep Learning in Semantic Segmentation of Rust in Images," in *ICSCA 2020: Proceedings of the 2020 9th International Conference on Software and Computer Applications*, 2020, pp. 129–132.
19. I. Katsamenis, E. Protopapadakis, A. Doulamis, N. Doulamis, and A. Voulodimos, "Pixel-Level Corrosion Detection on Metal Constructions by Fusion of Deep Learning Semantic and Contour Segmentation," in *International Symposium on Visual Computing (ISVC)*, 2020, pp. 160–169.
20. E. L. Bianchi, N. Sakib, C. Woolsey, and M. Hebdon, "Bridge Inspection Component Registration for Damage Evolution," *Structural Health Monitoring*, pp. 1–24, 2022.
21. T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *Proceedings of the 37th International Conference on Machine Learning*, ser. ICML'20. JMLR.org, 2020.