PROCEEDINGS OF SPIE

SPIEDigitalLibrary.org/conference-proceedings-of-spie

Damage segmentation using small convolutional neuronal networks and adversarial training methods on low-quality RGB video data

Kolja Hedrich, Lennart Hinz, Eduard Reithmeier

Kolja Hedrich, Lennart Hinz, Eduard Reithmeier, "Damage segmentation using small convolutional neuronal networks and adversarial training methods on low-quality RGB video data," Proc. SPIE 12019, AI and Optical Data Sciences III, 1201902 (2 March 2022); doi: 10.1117/12.2610123

SPIE.

Event: SPIE OPTO, 2022, San Francisco, California, United States

Damage segmentation using small convolutional neuronal networks and adversarial training methods on low-quality RGB video data

Kolja Hedrich[a], Lennart Hinz[a], and Eduard Reithmeier[a]

[a]Leibniz University Hannover, Institute of Measurement and Automatic Control, An der Universität 1, 30823 Garbsen, Germany

ABSTRACT

Within the aviation industry, considerable interest exists in minimizing possible maintenance expenses. In particular, the examination of critical components such as aircraft engines is of significant relevance. Currently, many inspection processes are still performed manually using hand-held endoscopes to detect coating damages in confined spaces and therefore require a high level of individual expertise. Particularly due to the often poorly illuminated video data, these manual inspections are susceptible to uncertainties. This motivates an automated defect detection to provide defined and comparable results and also enable significant cost savings. For such a hand-held application with video data of poor quality, small and fast Convolutional Neural Networks (CNNs) for the segmentation of coating damages are suitable and further examined in this work. Due to high efforts required in image annotation and a significant lack of broadly divergent image data (domain gap), only few expressive annotated images are available. This necessitates extensive training methods to utilize unsupervised domains and further exploit the sparsely annotated data. We propose novel training methods, which implement Generative Adversarial Networks (GAN) to improve the training of segmentation networks by optimizing weights and generating synthetic annotated RGB image data for further training procedures. For this, small individual encoder and decoder structures are designed to resemble the implemented structures of the GANs. This enables an exchange of weights and optimizer states from the GANs to the segmentation networks, which improves both convergence certainty and accuracy in training. The usage of unsupervised domains in training with the GANs leads to a better generalization of the networks and tackles the challenges caused by the domain gap. Furthermore, a test series is presented that demonstrates the impact of these methods compared to standard supervised training and transfer learning methods based on common datasets. Finally, the developed CNNs are compared to larger state-of-the-art segmentation networks in terms of feed-forward computational time, accuracy and training duration.

Keywords: Semantic segmentation, CNN, adversarial methods, GAN, damage inspection, endoscopic inspection, semi-supervised learning, U-Net, transfer learning

1. INTRODUCTION

The usage of Convolutional Neural Networks (CNN) for segmentation tasks is nowadays a common approach. Especially for visual tasks, CNNs provide extensive possibilities to automate such processes. This includes the proposed automated damage detection in aircraft engines presented in this study. In this process, an inspection of coating areas is carried out by post-processing endoscopic videos. These coating areas are located in confined spaces, which demand the usage of hand-held endoscopes. Currently, this is often performed by an inspector in form of a subjective visual quantification of the coating damages in the videos. Here, the resulting videos of the endoscope are utilized to provide an overview of the coating area and to identify the damage areas. The stream consists of 2D RGB image data and is afterwards available as a compressed video file. Due to this compression and bad environmental conditions (e.g. lighting, camera position) caused by the endoscope applied in the confined space, the videos are of low quality. To directly apply the methods presented in this study, all

Further author information: (Send correspondence to Kolja Hedrich)
Kolja Hedrich: E-mail: kolja.hedrich@imr.uni-hannover.de, Telephone: +49 (0)511 762 5396

AI and Optical Data Sciences III, edited by Bahram Jalali, Ken-ichi Kitayama, Proc. of SPIE Vol. 12019, 1201902
© 2022 SPIE · 0277-786X · doi: 10.1117/12.2610123

following investigations are based on the single frames of these video files. The actual hardware configuration and the inspection process regarding the handling of the endoscope remain unchanged.

The automation of the coating inspection comprises a semantic segmentation of the damage areas and will be tackled with a typical encoder-decoder structure (U-Net[1]). The wide range of damage occurrences requires developed neural networks that are well generalized. As a basic approach, supervised learning can be applied to train the networks for this segmentation task. But the use case in this work evokes two main challenges. On the one hand, the networks have to be small in terms of data memory size and must provide a fast feed-forward computational time to tackle the demands of a hand-held application in field. On the other hand, as in many CNN applications, annotated images require elaborate efforts. As stated before, the inspection process remains unchanged, resulting in a large amount of video data available from the past. The videos mainly differ in terms of the lighting conditions, surface structures, damage colors and shapes. Since therefore significantly more data is available than can be manually annotated, this leads to a lack of broadly divergent annotated image data (domain gap). In order to close this domain gap respectively utilize these unannotated images, semi-supervised learning methods will be presented in this work. For this three approaches are compared, which are a basic supervised learning method, (1) the training of a generative adversarial network (GAN[2]) to provide good initial weights for the encoder of the used U-Net and (2) a curriculum learning approach using a GAN to generate synthetic annotated image data, which can be used to perform further training of the used U-Net.[3] Since the U-Net is customized regarding its size, no suitable initial weights or pre-trained networks are available. Therefore, in other words, the proposed approaches have the goal to gain such good initial weights for the encoder (method 2 and 3) and decoder (method 3).

The utilization of GANs to improve supervised learning of basic U-Nets has already been examined in different ways. *Majurski et al.* propose the transfer of discriminator weights to initialize an U-Net applied on grayscale cell images.[4] In the work of *Ghassemi et al.*[5] and *Rohrer et al.*,[6] classification tasks are improved with this same semi-supervised method. Here, consequently most of the network could be initialized using the discriminator of the GAN. The approach of using GANs to extend an existing annotated dataset is also realized in various works. *Choi et al.* use GANs for data augmentation, which utilizes unannotated image data. Another approach is proposed by *Benjdira et al.*, where a GAN architecture is used to translate a dataset from one domain to another to gain a new synthetic dataset, which can be used for transfer learning.[7]

In this work, we handle RGB image data in small patches with very small U-Net structures regarding their filter channel sizes. Furthermore, in method (1) we only use the discriminator weights to initialize the encoder of the used U-Net. Method (2) includes the generation of a whole new annotated synthetic dataset using noise input vectors and a trained GAN generator. The dataset is then used for transfer learning of an U-Net.

2. MATERIAL AND METHODS

2.1 Datasets

The used image data gained from the hand-held endoscopes is characterized by inhomogeneous lighting conditions, which evoke significant noise and pixel intensity gradients over the image. The endoscope provides video files with a color depth of 24 bit (true color) and an inevitable MP4-compression.[8] The image data used for later dataset creation is gathered from the video frames by selecting a grid of patches with the resulting image size of 64x64 pixels (example images shown in Figure 3, left). This is done without any rescaling or downsampling. Hence, compression artifacts caused by the MP4-compression can cover a significant part of a single image patch. Also, the inhomogeneous lighting leads to a different mean pixel intensity for each image of a single frame. The available videos differ in lighting conditions, camera positions, the inspected coating structure and color as well as the damage characteristics. These environmental properties do not change as much over the frames of a single video. Therefore, the partitioning of the images of the different videos between the training, validation and test datasets is of particular importance and considerably influences the potential performance. For this, 25% of the videos are used for testing and 75% are assigned to training and validation. The training dataset is then built based on a defined proportion (between 3% - 50%) of the available video frames. With the same proportion, the validation dataset is built using the second half of the video frames, thus the datasets have the same size. This image data split regarding a single video is carried out for all assigned videos. The described data distribution

results in dataset sizes of approx. 200-5000 single images for training and validation each depending on the defined proportion and approx. 4000 images for testing, which are constant for all methods and experiments.

For the training and validation dataset a data augmentation is set up. This includes, next to noising and blurring, on the one hand geometric augmentations, such as flips, crops and light affine transformations. This implies mostly rotations and translations. On the other hand, color space transformations are applied, such as change in contrast and single color intensities. All these augmentations are applied in a random order and with random probabilities.

2.2 Experimental settings

The conducted experiments regarding the utilization of GANs as well as the training pipelines with U-Nets are realized with the *TensorFlow 2* framework on Python. The training and testing of large state-of-the-art networks for comparison, in turn, was implemented with the *PyTorch* framework and the *segmentation models pytorch* repository by *Yakubovskiy et al.*[9] The frameworks are configured to be fully deterministic for all experiments. This includes the seeding of all used third party libraries, e.g. for distributing the videos to the respective datasets, and applying the data augmentation methods in terms of order and probability. The experiments are conducted with 20 repetitions to achieve a uniform distribution, respectively enough combinations of the videos sorted to training, validation and testing. It is ensured that the test dataset is never seen by the respective trained network. Also, in a single repetition all datasets are exactly the same for all evaluated networks.

The output of the implemented networks is set to a binary mask (by thresholding the output values), since only two classes are of relevance: intact coating and damage area. The metric used for evaluation is the Dice similarity coefficient (or F-1 score) applied on the binary masks.[10] All experimental results are measured with the Dice calculated on all images in the test dataset using the best model in a training process. The best model is selected by the results gained from the validation dataset.

2.3 Segmentation networks

In this work, only U-Nets, similar to the original network presented by *Ronneberger et al.*, are used to tackle the segmentation tasks.[1] There are mainly two differences to the original U-Net. First, batch normalization is added to the structure. Second, the number of channels in each down-sampled depth is parameterized and selected to be significantly smaller during the course of the experiments. The depth itself, the order of all layers and number of convolutional layers still corresponds to the original paper.[1] The training process is executed using an Adam optimizer[11] with a default learning rate of $3 \cdot 10^{-4}$ and the categorical cross entropy as the loss function. Additionally an early stopping criterion is implemented to abort the training process if there is no further improvement in validation accuracy.

As a large state-of-the-art segmentation network the U-Net++[12] with a RegNetY-8.0GF[13] encoder is selected, since it performed best under several tested state-of-the-art networks on the available dataset. The whole network has 56,493,086 parameters and is initialized with ImageNet weights.

2.4 Generative adversarial network (GAN)

The GANs used in this work are mainly designed by the design rules for DCGANs.[14] The generator and discriminator for method (2) are exactly designed by these rules, since they have to fulfill the main purpose of the adversarial training and generation of synthetic data. For method (1) the generator network stays the same in contrast to the discriminator, which is replaced with the encoder of the U-Net mentioned before. This encoder violates most of the design rules. Nevertheless this is done, since in method (1) the weights have to be transferred from this discriminator to the U-Net encoder. To ensure a correct behavior of the encoder as a discriminator, an additional convolutional layer is appended to the encoder. The resulting discriminator used in the experiments in Section 3 has then only 75,461 parameters instead of the 2,766,657 parameters (discriminator DCGAN). All GANs are trained using DRAGAN[15] as the gradient penalty scheme and the binary cross entropy as the loss function. In each training step both discriminator and generator are updated by an Adam optimizer initialized with the default learning rate.

2.5 Proposed methods

The two proposed methods both utilize GANs to improve the training of an U-Net with sparse annotated image data. In both methods the identical U-Net should be improved with the same amount of annotated image data available. The only difference here, is the additional extensive usage of unannotated image data in the semi-supervised method (1). The overall goal of both methods is to still achieve high accuracies while decreasing the amount of image data used for training.

2.5.1 Transfer of discriminator weights

This first method utilizes the broadly available unannotated image data to tackle the domain gap. The method consists of an adversarial training with unannotated RGB image data of a generator and a discriminator, which resembles the encoder of the later trained U-Net. Since there is enough unannotated image data available, no data augmentation is applied here. In this method, the generator is solely used for the adversarial training. The weights of the discriminator in turn are transferred to the encoder of the U-Net, while the initial decoder weights are set randomly. Finally, a supervised training of this initialized U-Net is conducted with the sparse annotated image data for training. Due to the utilization of the unsupervised domain using the implemented GAN, this first method is a semi-supervised approach. For clarity, this method will be referred to as *TL-encoder* in the following.

2.5.2 Generation of synthetic data

The second proposed method utilizes a GAN to generate a synthetic dataset. However, this means that only annotated data can be used for the conducted training of the GAN and U-Net. The approach is resembling the method of curriculum learning.[3] For this, iterative trainings are conducted for both CNNs. First, a GAN is trained with image data from only few videos of the assigned training data. Subsequently, the GAN is trained with the available full training dataset. The output of the generator is here a four channel image consisting of an RGB image and a ground truth as a binary mask. Examples of this synthetic annotated image data can be seen in Figure 3 on the right. With this output of the generator, 5,000 synthetic annotated images are generated by passing random vectors to the generator. These images are then used to train a randomly initialized U-Net. Afterwards, the U-Net is again trained with the available training dataset containing real annotated image data. In this method the training data is augmented for the training of the U-Net and non-augmented for the adversarial training. Since only the supervised domain is used, this method is still a supervised method. The utilization of the GAN resembles a kind of data augmentation, which is applied in a first training run followed by a training with real image data. For clarity, this method will be referred to as *TL-synthetic* in the following.

3. EXPERIMENTAL RESULTS

This section covers the implementation and results of all conducted experiments. First, a supervised training pipeline for U-Nets with different numbers of filter channels is evaluated. This is done to select a suitable U-Net for the necessary scope of application and to give an overview of network capabilities regarding different sizes and numbers of (trainable) parameters. Second, the proposed methods are evaluated with the selected U-Net from the previous experiment and compared to a basic supervised training. Here, the results are displayed over the proportion of available video frames for training (see Section 2) to achieve an understanding about the effect of the proposed methods regarding the training dataset size. Finally, a large state-of-the-art model is trained and the results are compared to the best method from the previous experiments.

3.1 Identification of basic network parameter

In this experiment, the number of filter channels for all convolutional layers in each depth is adjusted to vary the network size. The original U-Net, as stated in the original paper,[1] has the following number of filter channels for each depth starting from highest to lowest depth: 64, 128, 256, 512, 1024. For simplicity, this setting is defined here as a number of root channels equal to 1024. In the first experiment this number of root channels is decreased with a factor of $\frac{1}{2}$ until 32. The channels in each depth for the respective setting are always decreased as in the original U-Net (by a factor of $\frac{1}{2}$).

The results with 20 repetitions and a constant proportion of available video frames equal to 0.5 can be seen in Figure 1. The Figure displays a boxplot and a data point for each repetition and each setting. It can be observed that with an increase of the number of root channels, the overall median accuracy increases. Also, the size of the box decreases drastically, which indicates a much higher convergence certainty. Only the last step to 1024 root channels depicts a small drop in accuracy and box size, which could be caused by the already large network size. The reason for the lower outliers and whiskers at 64, 128 and 256 root channels equaling in the same accuracy of 0.667, is a failed convergence of the network. This behavior is evident in form of a network output, which is either a mask containing only damage labels or a mask consisting only of intact coating labels. The overall ratio of the damage area in the full dataset equals to approx. 0.33%. Thus, the network achieves an accuracy of 0.667 when only outputting a black mask (respectively only intact coating labels). This could be validated by examining example outputs of all networks achieving this specific accuracy value.

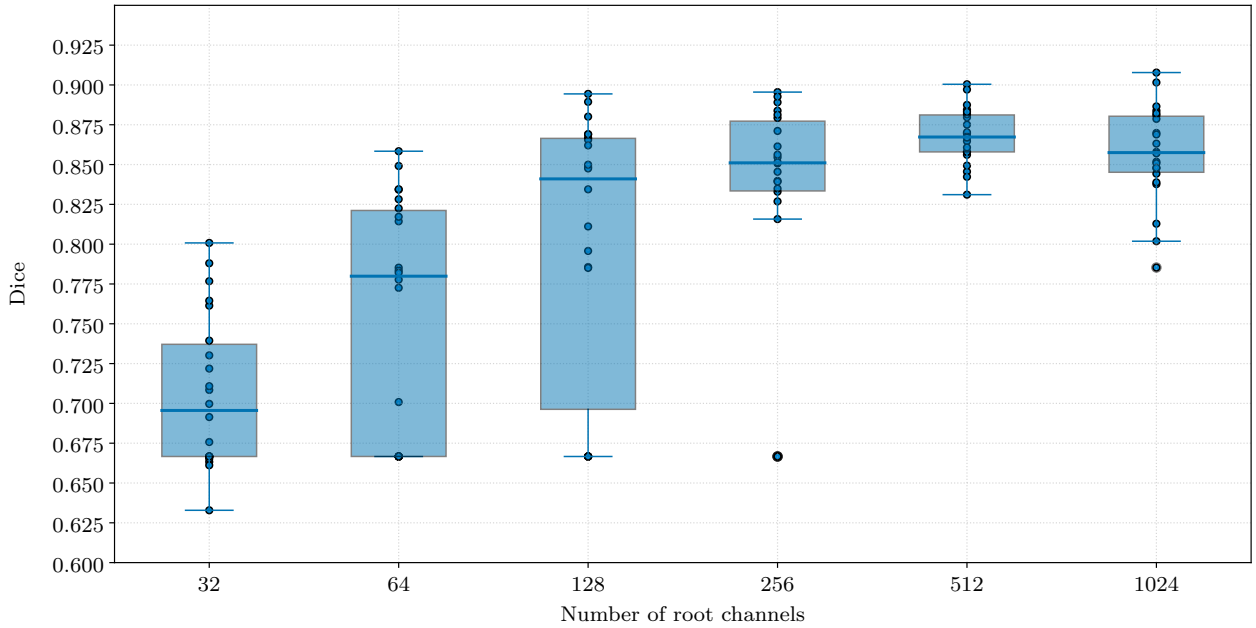

Figure 1. Boxplots comprising the training results of basic U-Nets with different numbers of root channels on the x-axis (non-linear). Each setting was repeated 20 times. The accuracy is measured by the dice coefficient with the best model on the test dataset.

In addition to the accuracy of the different U-Nets, the computational time (or feed-forward time) and the memory size of the respective model in *TensorFlow 2* are evaluated in Table 1. Here, for each network size the computational time regarding 100 feed-forward steps with a batch size of 12 is stated next to the total number of parameters and the memory size in MB. This is tested on an *NVIDIA Quadro GV100* graphics card. It can be seen that with a lower number of root channels the model size decreases approximately proportionally. This refers to the size in graphics memory, the disk space decreases significantly more. Further, the feed-forward time is remarkably lower at least for the first down-scaling of the model. For smaller U-Nets, 256 root channels and lower, there is only a marginal difference.

Since the aim of the whole investigation in this work is to develop a network, which will be applied in a hand-held application in field, the network number of root channels should be selected as small as possible to gain a fast feed-forward time and a small model size. Therefore, a U-Net with 64 root channels (namely following number of channels in each depth: 4, 8, 16, 32, 64) is selected. For this size, the accuracy is reduced compared to the larger networks but it can still be assumed that a correct convergence occurs in most runs. This cannot be safely presumed for the setting of 32 root channels. The goal in the following experiments is now, to improve the reduced accuracy with 64 root channels and prevent such failed convergences (see Figure 1, lower whisker at 64 root channels).

Table 1. Computational time (feed-forward time) of the used models with a batch size of 12, 20 warm-up steps and 100 timed steps. Tested on an *NVIDIA Quadro GV100*.

Model	Time (s)	Number of parameters	Size (MB)
U-Net (root channels: 1024)	28.30	28,095,522	378.4
U-Net (root channels: 512)	13.41	7,031,826	163.4
U-Net (root channels: 256)	9.34	1,761,930	75.7
U-Net (root channels: 128)	8.62	442,470	36.7
U-Net (root channels: 64)	8.03	111,612	18.4
U-Net (root channels: 32)	7.95	28,401	9.5

3.2 Experiments regarding the proposed methods

The following experiment compares the proposed methods with a supervised learning of a basic U-Net. For this, the U-Net chosen in the last subsection with 64 root channels is used. The results can be seen in Figure 2, where the boxplots regarding all three approaches over the proportion of training data is displayed. Here, 20 repetitions with different seeds were used, which causes some stochastic fluctuations between the experiments. Within an experiment, in each repetition, the stochastic conditions of the single configurations are congruent to each other by using the described seeding.

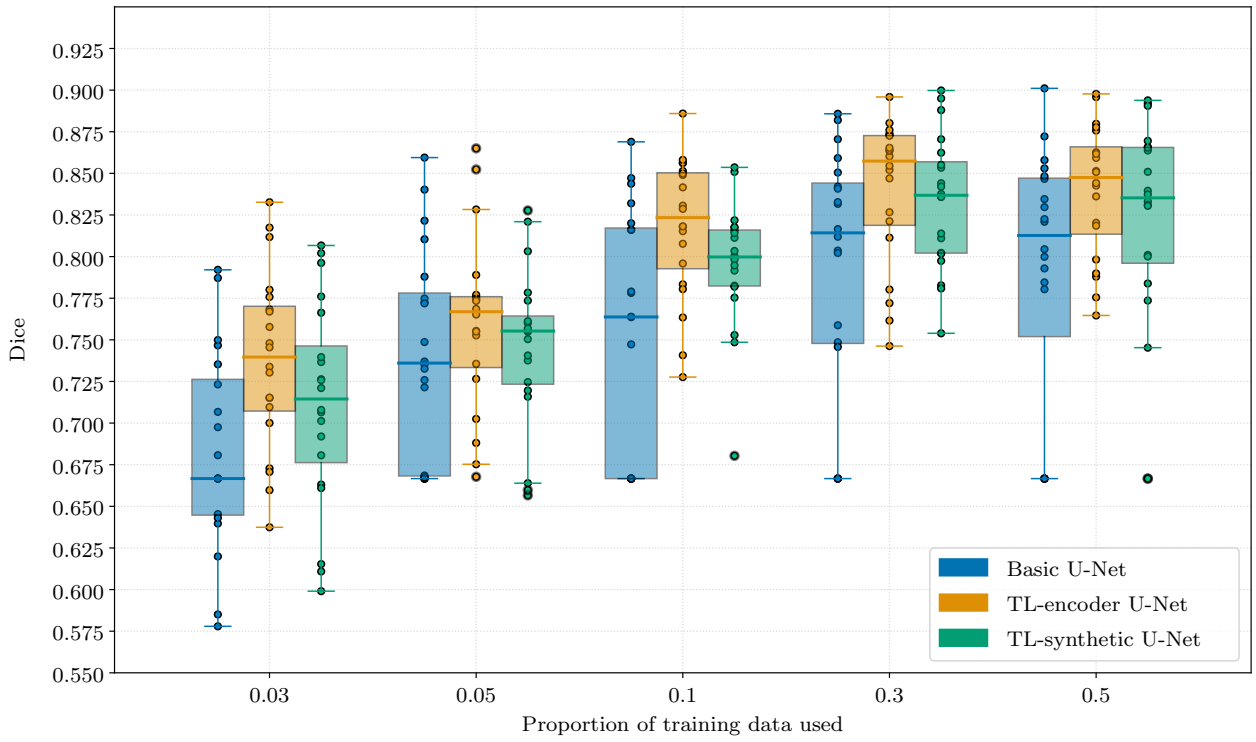

Figure 2. Boxplots comparing the basic U-Net, the TL-encoder U-Net and the TL-synthetic U-Net. All final U-Nets have 64 root channels. The x-axis is not linear and states different proportions of images available per video for training. The accuracy is measured by the Dice coefficient with the best model on the test dataset.

Several characteristics can be seen in the results. First, proposed method (1) has a higher median accuracy at all training dataset proportions (Figure 2, TL-encoder U-Net). This can especially be observed with the lowest

proportion of 3%. At proportions of 10% and higher, the method still achieves higher median accuracies but most of all prevents failed convergences. Even at the highest proportion of 50%, the basic U-Net has five failed convergences with an accuracy of 0.667 (lower outlier). These failed training procedures (resulting in only black network outputs) do not occur with the proposed method (1) for proportions of 10% and higher. For example at 10%, the basic U-Net entails eight such failed trainings. A high overall convergence certainty is indicated by a small interquartile range of the respective boxplot. Therefore, with method (1), the convergence certainty can be improved for proportions higher than 3%, since the range is decreased here. For method (2), it can be observed that the median accuracy can be slightly improved for all training dataset sizes compared to the basic U-Net (Figure 2, TL-synthetic U-Net). Still, the improvement is smaller than with method (1). The overall convergence certainty measured by the interquartile range could be increased for proportions between 5% and 30 %. Also, failed convergences with an accuracy of 0.667 still occur with almost all proportions. Additionally, the range between upper and lower whisker could be decreased significantly for proportions $\geq 10\%$.

The overall training duration of all methods differs depending on whether a GAN training has to be performed and which size the training dataset is. For a basic supervised training with a training dataset proportion of 0.5, a mean duration of 117 minutes is required until the best model is achieved. The additional training of a GAN in method (1) requires approx. 353 minutes. For method (2), a GAN training, a generation of a synthetic dataset and a training of an U-Net with the synthetic data is additionally needed. Therefore, this method requires another approx. 518 minutes. This is tested without test cycles on a *NVIDIA Quadro GV100*.

Example images and segmentations regarding the results of this experiment can be seen in Figure 3. Here, two original images from the test dataset are displayed in the left column. Next, the according ground truth labeled by hand is shown. The results of the basic U-Net and the U-Net trained by method (1) are compared. It can be seen that for the upper test image the output from the basic U-Net misses certain parts of the damage area. The output of the TL-encoder U-Net is mostly correct. The three columns on the right depict the outputs from method (2). Here, the synthetic RGB data and respective ground truth produced by the trained generator are displayed. The generator is trained with an training data proportion of 30%. Except for some artifacts, the damages appear mostly realistic. The ground truth, on the other hand, consists partly of too extensive areas labeled as damage. On the right, the output of a randomly initialized U-Net, trained on the synthetic dataset, is shown. This U-Net would later be trained on the real training dataset in method (2).

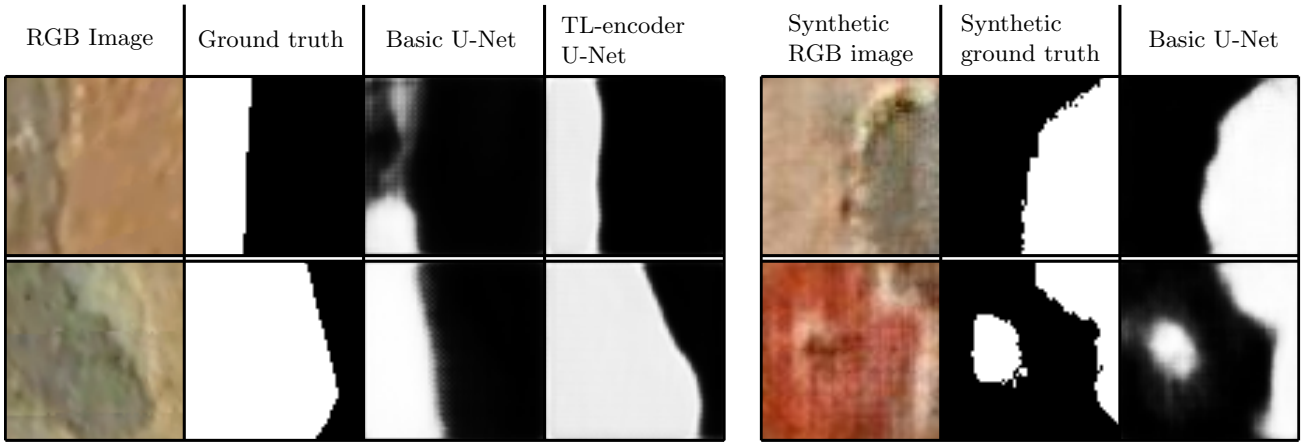

Figure 3. Example images on the left depicting the original image data, the manually labeled ground truth, the segmentation result of a basic U-Net trained on 3% of the available training data and a TL-encoder U-Net trained on the same dataset size. On the right, example results of the generation and utilization of synthetic data are shown. The right column comprises the segmentation result of a basic U-Net pre-trained on this synthetic dataset. The synthetic dataset was trained with 30% of the available training data. The image data displayed has a resolution of 64x64 pixels.

3.3 Comparison to a large state-of-the-art network

In the last experiment, the proposed methods are compared to a large state-of-the-art network. The described network, a U-Net++ with a RegNetY encoder, is set up in the *PyTorch* framework. Nevertheless, the training is

performed using the same training dataset properties as the previous experiments. Here, the video file distribution is executed with the identical seeds. It could not be assured that other algorithms and third party libraries work identical in this training pipeline.

Table 2. Comparison between the TL-encoder U-Net and a state-of-the-art segmentation network (U-Net++[12] & RegNetY[13]). The table displays the mean accuracy (μ) and standard deviation (σ) of the training results with 20 repetitions measured by the Dice.

Proportion of training data used	**TL-encoder U-Net**		**U-Net++ & RegNetY**	
	$\boldsymbol{\mu}$	$\boldsymbol{\sigma}$	$\boldsymbol{\mu}$	$\boldsymbol{\sigma}$
0.03	0.7208	0.0374	0.8865	0.0217
0.05	0.7406	0.0390	0.8890	0.0247
0.10	0.7798	0.0288	0.8838	0.0333
0.30	0.8163	0.0265	0.9009	0.0316
0.50	0.8095	0.0308	0.9085	0.0214

To achieve a reasonable comparison, the TL-encoder U-Net and the U-Net++ are compared over the proportion of training data used. The results can be seen in Table 2, where the same proportion sizes are examined as before. It is noticeable that the U-Net++ achieves high accuracies with all tested proportions. The improvement in accuracy using a ten times bigger dataset is only approx. 1.5%, while the TL-encoder U-Net improves approx. 6-9%. A comparison of the absolute accuracy values is biased due to the different frameworks used for the compared networks (*TensorFlow*, *PyTorch*). Nevertheless, the U-Net++ achieves a significantly higher accuracy, which is approx. 10-16% better than the developed TL-encoder U-Net under the given conditions.

4. DISCUSSION

The focus of this study is on the improvement of training pipelines for small U-Nets to be applied with hand-held endoscopes in an industrial environment. The main objectives are two proposed methods, which utilize GANs to gain good initial weights for a supervised training. Further, these small networks are compared against a large state-of-the-art network.

The results in Section 3 depict the different weaknesses and benefits of the proposed methods. The transfer of discriminator weights to the encoder, method (1), successfully improves the overall accuracy regardless of the training dataset size. Especially for low dataset sizes, this method yields a significant increase in accuracy compared to a basic supervised training with random initial weights. Also, most failed convergences (as for the basic U-Net training) can be prevented (see Section 3). The overall convergence certainty could only be improved slightly. This is probably due to the final results of the supervised training, performed in all approaches, strongly depend on the distribution of the videos among the datasets. The transferred weights in method (1) only provide a good initial estimation to support convergence and in order to achieve higher accuracies in the supervised training. One main disadvantage of method (1), is the relatively long training duration due to the necessary training of the GAN. The generation of a synthetic dataset with an individually trained GAN, method (2), achieved mixed results. The median accuracy could only be improved slightly for all proportions. For lower dataset sizes, this could be due to a synthetic dataset of poor quality generated by a GAN, which only got few image data at low proportions. This could result in the U-Net learning unrealistic features and therefore lower the overall performance. For proportions of 10% and higher, the trained GAN outputs plausible synthetic data (see Figure 3, right), which leads to more advantageous initial weights of the pre-trained U-Net. This probably also led to the increased convergence certainty with proportions of 10% and higher. Nevertheless, the accuracy improvement at these larger datasets is only marginal compared to the TL-encoder U-Net. An explanation for the better results with method (1) could be the additional utilization of the unsupervised domain. Since this domain is significantly larger than the supervised domain, the TL-encoder U-Net probably starts with more

generalized weights, which therefore lead to better accuracies. Method (2), which uses only proportions of the supervised domain for the generation of initial weights, could therefore have negative effects such as overfitting.

The examination of a large state-of-the-art network with the same training settings indicates that for such large networks the size of the training dataset does not matter as much as for small networks. Based on post-investigations, it was observed that only when excluding whole video files, instead of decreasing the number of used images per video, the accuracy of the applied large network decreases significantly.

5. CONCLUSION AND FUTURE WORK

In this study, small encoder-decoder networks for coating damage segmentation tasks were investigated. The application with hand-held endoscopes in field demands for small and fast networks, since the inspection operations usually have to be carried out with portable and less powerful computers respectively graphic cards. The aim of this work was to improve training methods of the implemented U-Nets in terms of accuracy and convergence certainty using GANs. Because a large unsupervised domain is available and annotated data is sparse, a first method was proposed to utilize this unannotated image data by training a GAN and transferring the discriminator weights to the U-Net encoder. In a second method, a trained GAN is used to generate synthetic annotated image data to pre-train a U-Net. Thus, both methods aim for a good initialization of the U-Net weights, to ensure an efficient training with the small training datasets. Especially the first method increases successfully the overall accuracy and convergence certainty. The second method works best on larger datasets under the given conditions. Still, such small U-Nets cannot compete with large state-of-the-art networks in accuracy. The overall training time including the training of the respective GAN exceeds the duration of a basic supervised training drastically. Nevertheless, the improvement in accuracy and convergence certainty are beneficial for the presented application.

However, the proposed methods could be extended or combined to further improve the training of such U-Nets. A main approach could be an adequate initialization of the U-Net decoder in method (1), since the weights are currently initialized random. Here, a combination with method (2) or the development of a similar GAN generator for weight transfer is conceivable. Eventually, the development of more efficient network structures based on state-of-the-art network blocks could lead to better results, while keeping the network small.

ACKNOWLEDGMENTS

The underlying project of this conference contribution was funded by the German Federal Ministry of Education and Research as part of the Aviation Research and Technology Program of the Niedersachsen Ministry of Economic Affairs, Employment, Transport and Digitalisation. The project is carried out in cooperation with the MTU Maintenance Hannover GmbH.

REFERENCES

[1] Ronneberger, O., Fischer, P., and Brox, T., "U-Net: Convolutional networks for biomedical image segmentation," in [*International Conference on Medical image computing and computer-assisted intervention*], 234–241, Springer (2015).

[2] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y., "Generative adversarial nets," *Advances in neural information processing systems* **27** (2014).

[3] Lotter, W., Sorensen, G., and Cox, D., "A multi-scale CNN and curriculum learning strategy for mammogram classification," in [*Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*], 169–177, Springer (2017).

[4] Majurski, M., Manescu, P., Padi, S., Schaub, N., Hotaling, N., Simon, C., and Bajcsy, P., "Cell image segmentation using generative adversarial networks, transfer learning, and augmentations," (2019).

[5] Ghassemi, N., Shoeibi, A., and Rouhani, M., "Deep neural network with generative adversarial networks pre-training for brain tumor classification based on MR images," *Biomedical Signal Processing and Control* **57**, 101678 (mar 2020).

[6] Rohrer, T. and Kolberg, J., "GAN pretraining for deep convolutional autoencoders applied to software-based fingerprint presentation attack detection," *arXiv:2105.10213* (2021).

[7] Benjdira, B., Bazi, Y., Koubaa, A., and Ouni, K., "Unsupervised domain adaptation using generative adversarial networks for semantic segmentation of aerial images," *Remote Sensing* **11**(11), 1369 (2019).
[8] "ISO/IEC 14496-14:2003: Information technology — Coding of audio-visual objects — Part 14: MP4 file format," tech. rep., International Organization for Standardization, Geneva, CH (2003).
[9] Yakubovskiy, P., "Segmentation models PyTorch." https://github.com/qubvel/segmentation_models.pytorch (2020).
[10] Dice, L. R., "Measures of the amount of ecologic association between species," *Ecology* **26**(3), 297–302 (1945).
[11] Kingma, D. P. and Ba, J., "Adam: A method for stochastic optimization," *arXiv:1412.6980* (2014).
[12] Zhou, Z., Siddiquee, M. M. R., Tajbakhsh, N., and Liang, J., "UNet++: A nested U-Net architecture for medical image segmentation," in [*Deep learning in medical image analysis and multimodal learning for clinical decision support*], 3–11, Springer (2018).
[13] Radosavovic, I., Kosaraju, R. P., Girshick, R., He, K., and Dollár, P., "Designing network design spaces," (2020).
[14] Radford, A., Metz, L., and Chintala, S., "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv:1511.06434* (2015).
[15] Kodali, N., Abernethy, J., Hays, J., and Kira, Z., "On convergence and stability of GANs," *arXiv:1705.07215* (2017).