Synthetic training data bias in instance segmentation algorithms

Lena R. Schreiber[a], Yannick E. Tarant[a], and Kai Franke[a]

[a]German Aerospace Center (DLR), Institute for the Protection of Terrestrial Infrastructures, Rathausallee 12, 53757 Sankt Augustin, Germany

ABSTRACT

Protecting infrastructures, particularly buildings, requires the development of automatic detection and mapping technologies for essential components such as pipes in fire extinguishing systems. Despite machine learning advancements in instance segmentation, acquiring extensive, well-annotated real data remains challenging. This leads to the exploration of synthetic images as an alternative solution. This study addresses the limitations of synthetic training data, which can lack realistic features present in real-world images, resulting in biased models and decreased performance. Leveraging Unreal Engine 5 (UE5), a synthetic dataset resembling real-world data from a specific scene is generated. Creating such realistic worlds is time-consuming, so varying domain randomization levels and preprocessing using image filters are explored. With different training set combination, consisting of various distribution of real, synthetic and augmented data, multiple models are trained based on Mask R-CNN and YOLO. After the training phase, an optimization procedure is applied to each model, enabling a comparative analysis of pipe instance segmentation quality for different algorithms based on the composition of the training set. The findings shed light on the efficacy and potential risks of employing synthetic data for training various instance segmentation models. This study provides valuable insights into mitigating challenges associated with data limitations in the training of state-of-the-art neural networks.

Keywords: Critical infrastructures Training efficiency Synthetic Data Domain randomization Automated detection Instance segmentation Machine learning Mask R-CNN YOLO Unreal Engine 5

1. INTRODUCTION

Object detection and instance segmentation are fundamental tasks in computer vision, where the goal is to identify and localize objects within an image. Advanced deep learning models such as YOLO (You Only Look Once)[1] and Mask R-CNN (Region-based Convolutional Neural Networks)[2] have demonstrated remarkable capabilities in performing these tasks. These models rely heavily on large amounts of annotated training data to achieve high accuracy and generalization capabilities. The quality and diversity of the training datasets directly influence the performance of the models.[3] However, acquiring real-world annotated data is often a significant challenge due to high costs, time consumption, privacy concerns, and the difficulty in capturing rare or hazardous environments.

As a potential solution, synthetic data generation has emerged as an alternative or supplementary approach to real data. It allows for the creation of large datasets with precise annotations at a lower cost.[4] However, synthetic data introduces its own set of challenges, primarily the differences between synthetic and real-world data that can reduce the effectiveness of models when applied to real-world situations. Bridging this reality gap is critical for leveraging synthetic data effectively.[5]

Domain randomization is a powerful technique designed to bridge this reality gap.[6] By systematically varying the parameters of synthetic data generation, domain randomization exposes models to a broad spectrum of visual variations during training, enhancing their robustness and generalization capabilities. This chapter explores how domain randomization, particularly when implemented using advanced tools such as Unreal Engine 5, can help mediate this shortcoming. We will discuss the principles of domain randomization and the impact of training data diversity on the performance of deep learning models in real-world applications.

Further author information: (Send correspondence to Lena R. Schreiber)
Lena R. Schreiber: E-mail: lena.schreiber@dlr.de, Telephone: +49 2241 20148 37
Yannick E. Tarant: E-mail: yannick.tarant@dlr.de, Telephone: +49 2241 20148 86
Kai Franke: E-mail: kai.franke@dlr.de, Telephone: +49 2241 20148 52

Artificial Intelligence for Security and Defence Applications II, edited by Henri Bouma, Radhakrishna Prabhu, Yitzhak Yitzhaky, Hugo J. Kuijf, Proc. of SPIE Vol. 13206, 132060L · © 2024 SPIE
0277-786X · doi: 10.1117/12.3030822

2. MATERIALS AND METHODS

In this study, we introduce a dataset mixing method that combines RGB images with synthetic data, incorporating various levels of domain randomization. We evaluate the effectiveness of our dataset mixing strategy using two different instance segmentation algorithms, with the Jaccard score[7] serving as the primary metric for assessment.

2.1 Data Acquisition

Data acquisition is a crucial step in the development and training of machine learning models, particularly in the field of computer vision. The effectiveness of these models heavily relies on the quality, diversity, and quantity of the data they are trained on. We used real-world and synthetic data, supplemented by data augmentation and domain randomization techniques.

2.1.1 Real-World Data

The real-world object detection images were obtained in the basement of an industrial complex, specifically where a pipe structure for a fire extinguishing system is located. A set of 399 RGB images was taken using a Canon EOS 7D SLR camera, originally at a resolution of 6000x4000 pixels. The dataset includes around 9000 instances of pipe segments, although their distribution across the images is not even. The data is split into three separate image sets, a training set with 279 images, a validation set containing 80 images, and 40 images in the testing set. The software employed for creating ground truth data is Labelme.[8]

2.1.2 Synthetic Data

Unreal Engine 5 is a powerful game engine that can create highly realistic 3D environments and objects. By leveraging its capabilities, synthetic data can be generated with high fidelity. The engine allows for detailed control over scene parameters, enabling the creation of diverse and complex datasets that mimic real-world conditions closely.

Domain Randomisation Domain randomization (DR) involves the systematic variation of visual parameters in synthetic data generation. This technique helps models become more robust by exposing them to a wide range of variations during training. Parameters such as lighting, textures, object poses, and backgrounds are randomized, ensuring that the model learns to generalize across different scenarios.

Site Creation In order to generate synthetic data with a sufficiently high level of domain randomization, a procedural generator (site creation) is used for sites with pipe systems. The generator creates the site from a catalog of individual system components, which it can generate, arrange and connect semantically correctly. It is therefore initially able to generate walls, floors and ceilings that match each other. Pipe systems and lighting components are then generated and integrated into the environment. Finally, clutter objects are placed in the synthetically generated system to increase domain randomization. The generation process is modular thanks to the component approach so that, for example, different pipe systems or lighting can be tested within rooms that remain the same.

Colosseum Colosseum[9] simplifies the process of capturing screenshots and instance-segmented images directly from Unreal Engine environments. It's a tool designed for efficient image collection, ideal for tasks like documentation, quality assurance, and data generation for machine learning.

2.2 Training of Deep Learning Network

The labeled real and synthetic data is fed into both the YOLO and the Mask R-CNN object detection training algorithm, adjusting the model's weights to recognize the annotated classes. Subsequently, the generated models are compared to determine the optimal composition of real and synthetic data for training set quality.
The YOLO model is built in Pytorch, while the Mask R-CNN implementation from Matterport runs on Keras and Tensorflow frameworks. Details of the essential machine learning packages and their versions for this implementation are outlined in Table 1. All models, YOLO and Mask R-CNN, are trained for 200 epochs, followed by assessing the best created model found through analysis of the Jaccard Score.

Table 1: List of machine learning packages and their respective version

Package	Version for Mask R-CNN	Version for YOLO
Tensorflow-gpu	1.14.0	-
Keras	2.2.4	-
CUDA	12.1	12.1
Cudatoolkit	10.0.130	-
Cudnn	7.6.5	8.9.2.26
Torch	-	2.3.0
Torchvision	-	0.18.0
ultralytics	-	8.2.28

2.2.1 YOLOv8

The YOLOv8 model by Ultralytics[1] is the eighth version of the You Only Look Once (YOLO) deep neural network initially released in 2023. YOLOv8 instance segmentation includes multiple architectures of varying depths, all pre-trained on the COCO dataset.[10] For the intended purpose, the largest architecture is chosen to maximize the model's performance. For all trainings, standard augmentations were applied.

2.2.2 Matterport Mask R-CNN

Matterport offers a Python3 implementation of the Mask R-CNN algorithm built on Keras and Tensorflow. This model performs object segmentation using a deep learning network based on ResNet architecture and is accessible through a GitHub repository.[11] It detects objects in images by analysing both low-level features such as edges and corners, and subsequently higher-level features that provide semantic context, such as people, cars, and animals. Depending on the labelled objects, the Mask R-CNN algorithm can recognize a wide range of object types. The pre-trained model weights are trained on COCO dataset[10] using training data paired with their annotated ground truth. To streamline the training process, hyperparameters are tuned.

2.3 AI Tools

In the process of writing, ChatGPT was used to improve the English in some parts of the Materials and Methods chapter of the script.

2.4 Evaluation Methods

In order to make a statement about the effectiveness of the procedures discussed here, a quantitative measure must be found to describe both domain diversity and model quality.

2.4.1 Quantifying the Models Quality

The evaluation of the models is quantified by analysing the predicted masks on the test images. To quantify the detection quality of the mask prediction the Jaccard score is used.[7] It is based on the Intersection over Union (IoU) method, but applied to masks instead of bounding boxes. The ground truth masks and the predicted masks are compared pixel wise and the overlapping is measure as a value between 0 and 1. The Jaccard score is then defined per image rather than per instance, so that false positives can influence the score as well.

2.4.2 Quantifying the Training Set Diversity

To ensure the synthetic data generated is diverse enough, we measure its diversity using Truncated CLIP Entropy (TCE).[12] TCE quantifies the variety of data in a training set, utilizing the second-to-last layer of the pre-trained CLIP model. The desired image set is put through the network and the distribution at this layer is used to measuring the entropy of predictions, calculating the entropy of the truncated eigenvalues of the empirical covariance matrix of these encodings. This method assesses the distribution of predictions, capturing the diversity within the data. Higher TCE values indicate greater diversity, which is crucial for training models that generalize well to real-world scenarios.

2.5 Training Data Set

2.5.1 Training Set Compositions

Domain randomization is based on the idea that by introducing a wide range of variations in the synthetic training data, the model can learn to generalize better to real-world conditions. This technique involves randomizing different aspects of the synthetic environment, such as:

- Object Poses and Orientations: Altering the positions, rotations, and scales of objects.
- Lighting Conditions: Varying the number and position of light sources.
- Textures and Materials: Changing the textures and materials of objects and backgrounds.
- Clutter: Using different clutter objects.

For all training sets, we used the same room setup. In this room setup, we varied the number, size, and position of the pipes to be detected, thereby generating different room configurations. Additionally, we randomly placed clutter objects in various quantities and positions within the room. For all pipes, clutter, and backgrounds, we varied the materials and textures, ranging from simple smooth surfaces to reflective and textured materials in varying colors.

In this configured room, we employed the same number of light sources with identical intensity and position. clutter and variations in object placement introduced areas of light and shadow. Only for the training data of the *high DR* set did we additionally vary the number and positions of the light sources. We modified the materials and textures for the test sets *high DR* and *no clutter*. The *one material* and *low DR* sets consistently used the same textures and materials for their respective objects. In the *no clutter* test set, clutter objects were excluded.

Table 2: Training set compositions

Training set	Material	Clutter	Light	TCE (10 rooms)
low DR	one	-	x	14,810
one material	one	x	x	20,623
no clutter	all	-	x	28,461
high DR	all	x	variation	29,251

We created training sets with varying TCE values by using different levels of domain randomisation (Table 2).

2.5.2 Training set mixing strategy

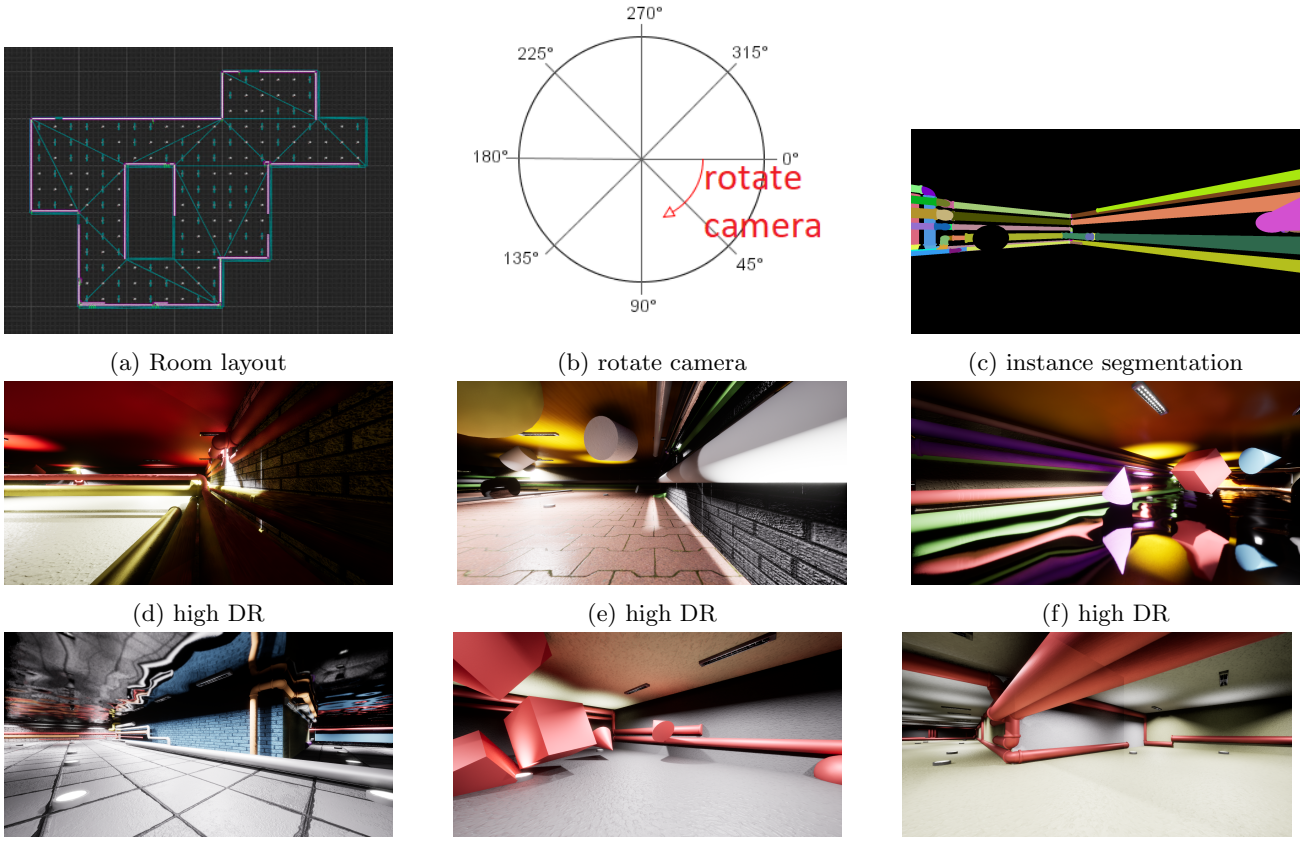
If only a limited amount of real data is available, the training set can be augmented with synthetic images. To evaluate the effectiveness and optimal ratios for mixing real and synthetic data, we combined 279 real images with varying numbers of synthetic images. We tested ratios starting from 0% synthetic images, progressively reducing the real data to as low as 3% (see figure 6). Additionally we compared the effects of mixing real images with synthetic images at various DR levels (see figure 7).

3. RESULTS

3.1 Levels of domain randomisation

3.1.1 Generating Training Data

Our approach for generating training data with varying levels of domain randomization for the training of neural networks is shown in fig. 1 .



(a) Room layout (b) rotate camera (c) instance segmentation

(d) high DR (e) high DR (f) high DR

(g) no clutter (h) one material (i) low DR

Figure 1: Levels of domain randomization

Figure 1 (a) displays the used room layout. With specific DR configurations we configured different rooms for each desired DR level. At 19 positions within this room we rotated the camera with 45° steps around itself (b) and took RGB (d-i) and instance segmented (c) images. By this we got 133 images from each room. By using multiple rooms we could gather a high number of images and additionally increase the TCE score for all training sets as shown in figure 2.

For the *low DR* set the TCE score settles under 15.000. The *high DR* set TCE score settles for just under 30.000 for 20 rooms. By using only one simple material for each object the TCE score decreases to under 21300. Neglecting additional clutter in the room decreases the TCE score slightly for lower numbers of rooms but even reaches a higher score for 20 rooms.

3.1.2 YOLO training

In the following section the results of the YOLOv8 training with various training sets are presented. Each model is evaluated by calculating the Jaccard Score on a test set consisting of 40 images from the real image's dataset.

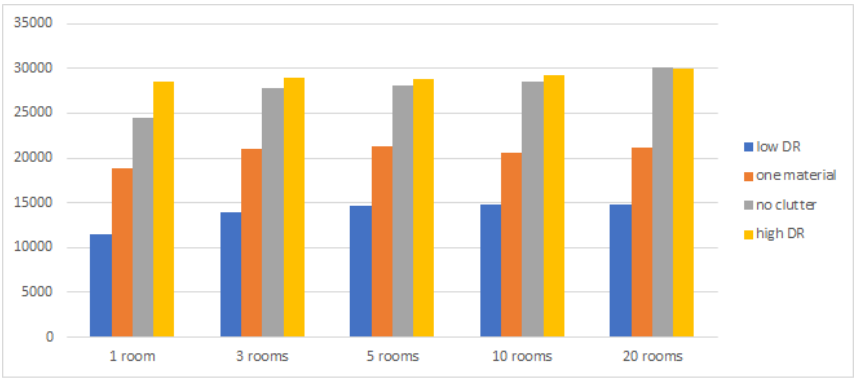


Figure 2: TCE score depending on number of different rooms. 133 images from each room.

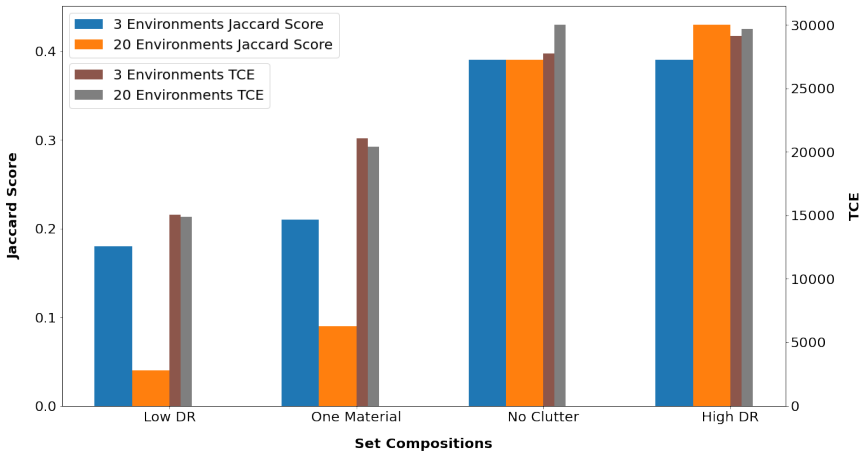


Figure 3: Jaccard Score depending on level of randomisation for 3 and 20 different environments

With each aforementioned level of domain randomisation, various models were trained. Figure 3 shows the TCE value of the training set and the Jaccard Score of the resulting model. For the models trained with the *low DR* set and the *one material* set, an increase in the number of environments leads to a slight decrease in TCE value and a heavy deterioration in performance on the test set. The images of three environments of the *no clutter* and *high DR* training sets exhibit a higher TCE than the other two. The YOLOv8 models of the *no clutter* set with 3 and 20 environments predict the same Jaccard Score on the test set. When increasing the number of rooms on the *high DR* set, we score a higher Jaccard Score.

Model quality correlations As can be seen in figure 2, the TCE value of the training set increases fast for the first couple of images, but more slowly the more images the dataset contains. In order to see if the model quality, trained with these image sets, behaves accordingly, figure 4 lists the TCE, Jaccard Score, and mean average precision values of the *high DR* set depending on the amount of rooms.

Starting with 1 room, the Jaccard Score, the TCE, the mAP50, and the mAP50-95 all show relatively low values. As the number of rooms increases, the evaluation metrics increase as well. The Jaccard Score of the prediction on the test set images starts with a value of 0.38 and has its peak at 0.46 for 60 rooms. The gain in performance is rather steady with exceptions at the number of rooms 2 and 60. However, the increase is not linear over the amount of rooms. After a strong initial increase, the increase slows down. Enlarging the data set

# of Rooms	TCE	Jaccard Score	mAP50	mAP50-95
1	28,725	0.38	0.208	0.102
2	29,247	0.41	0.212	0.105
3	29,106	0.39	0.209	0.101
4	29,392	0.4	0.220	0.103
5	29,429	0.4	0.205	0.098
6	28,966	0.41	0.233	0.112
7	29,201	0.42	0.230	0.111
8	29,168	0.42	0.225	0.111
9	29,513	0.42	0.234	0.112
10	29,309	0.42	0.246	0.116
20	29,689	0.43	0.226	0.114
30	29,925	0.43	0.234	0.122
40	30,178	0.43	0.242	0.125
50	30,193	0.44	0.266	0.147
60	29,902	0.46	0.253	0.134
70	30,020	0.43	0.264	0.139
80	30,084	0.44	0.258	0.139

Figure 4: All values for TCE, Jaccard Score, mAP50, and mAP50-95 for the *high DR* data set. (1 room = 123 images)

from 1 room to 7 rooms results in a Jaccard Score gain of 0.4, 50% of the total increase in performance, whereas the other 50% are achieved after increasing the room number to 60.

	Rooms	TCE	Jaccard Score	mA50	mAP50-95
Rooms	1.00	0.88	0.90	0.90	0.91
TCE	0.88	1.00	0.80	0.75	0.83
Jaccard Score	0.90	0.80	1.00	0.82	0.83
mA50	0.90	0.75	0.82	1.00	0.96
mAP50-95	0.91	0.83	0.83	0.96	1.00

Figure 5: Correlation matrix of the Number of Rooms, TCE, Jaccard Score, mAP50, and mAP50-95 for the *high DR* set.

This nonlinear behaviour is evident across all parameters. Figure 5 shows the correlation matrix of the five parameters. It can be seen that all variables correlate with a value over 0.75. As they are calculated very similarly, the two parameters with the strongest correlation of 0.96 are mAP50 and mAP50-95. With values of 0.88 to 0.91 the amount of rooms has high correlation to the other parameters. The calculation of the TCE requires significantly less resources than the training of a typical deep learning object detection network. Since the correlation between the here chosen evaluation methods and the TCE value is above 0.75, the metric could be used as pre-training evaluation to give an estimate over the quality of the data set, before processing computationally more intensive algorithms.

3.2 Training set mixing strategy

In this section we evaluate the mixing of the synthetic data with a real image training set.

3.2.1 YOLO training

Ratio of synthetic and real data Starting with the pure real training data set, we gradually add *high DR* images, shifting the ratio of real and synthetic images in the training set. The results are shown in the figure 6. The pure real data set covers 279 images and produces a model which achieves a Jaccard Score of 0.54, 0.09 higher than the best synthetic data set.

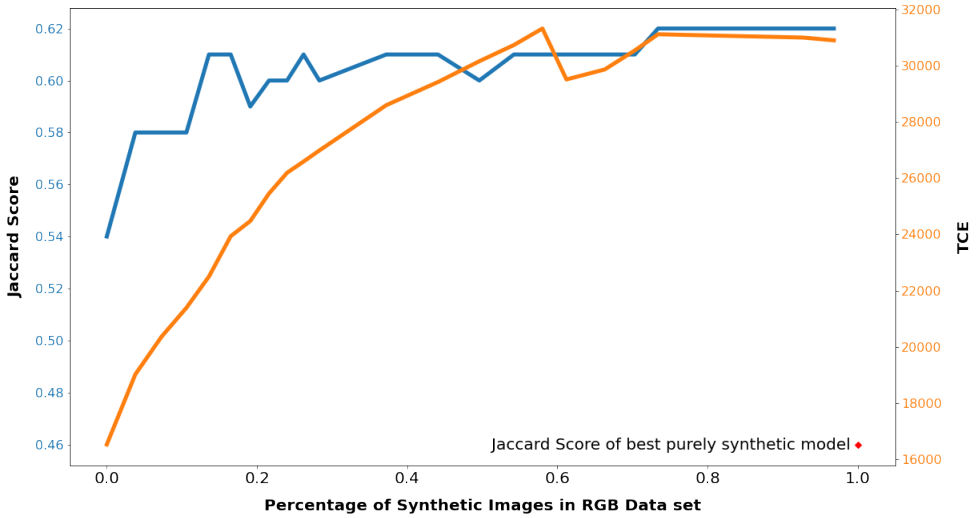


Figure 6: Jaccard Score of different real and synthetic data constellations.

The addition of *high DR* synthetic images increases the model quality instantly by 0.04. Increasing the number of added synthetic images creates models which score higher Jaccard Scores in average. The quality of our model does not drop, even if the relation reaches close to 100% synthetic data. As soon as there are no real images in the data set, the quality drops to 0.46.

Altering the quality of the synthetic data Adding synthetic data with different TCE levels, results in different model performances. Figure 7 shows the training results of real data sets mixed with synthetic data of different levels of DR alongside the according TCE values. All added synthetic image sets consist of 20 environments with 123 images each.

The TCE value of the real data is lower compared to the other data sets. All real data are collected from only one site and therefore, lack variation. However, since the data is more akin to the testing data, the model still performs well. Adding synthetic images which did not perform well on their own (see figure 3), results in a deterioration of the performance, compared to the pure real training set. Although an increased TCE value can be observed. The combination of real images and synthetic images of 20 different *no clutter* rooms results in an increase in Jaccard Score and TCE. Eventhough the TCE value of this combination is higher than the value of the mixed real and synthetic of *high DR* training set, the latter creates the best model of all different training set resulting in a Jaccard Score of 0.62.

3.2.2 Mask R-CNN training

To compare our results with YOLOv8 we use Mask R-CNN as a second comparable algorithm. By this we can show to which degree the resulting model is dependent on the computer vision algorithm and not the training set composition.

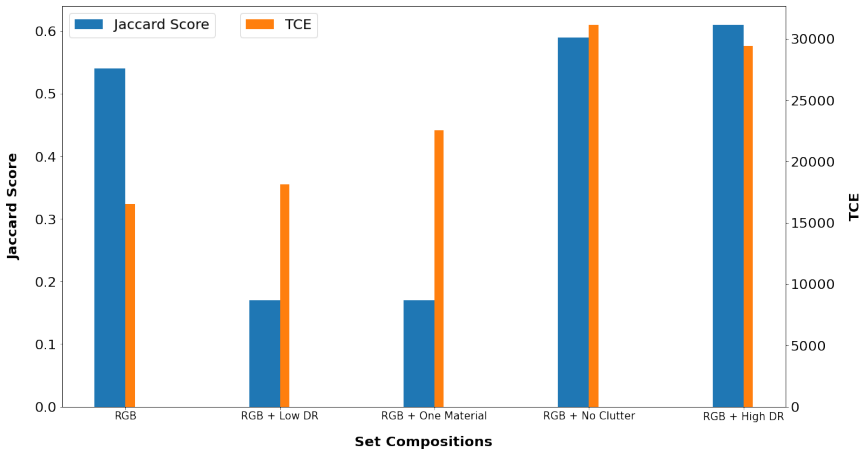


Figure 7: Jaccard Score and TCE of models trained with real data and synthetic data of different levels of domain randomised data

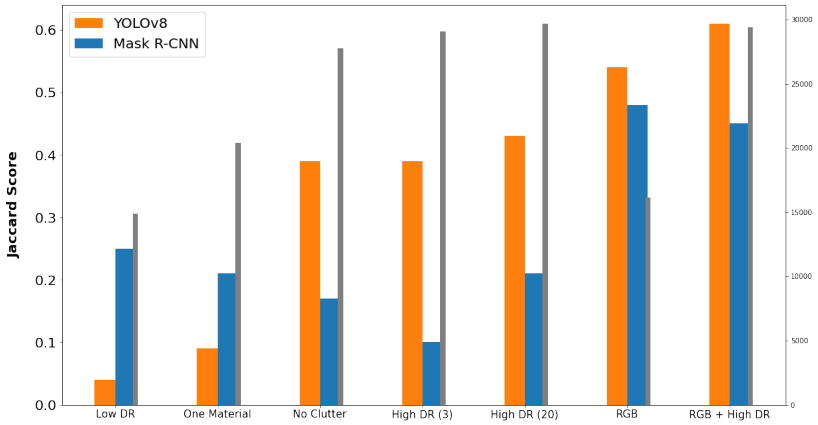


Figure 8: Jaccard Score of different training sets for Mask R-CNN and YOLOv8 models. Skinny gray bars show the TCE value

The results of the Mask R-CNN and YOLO trainings are shown in figure 8. The training sets with lower TCE values (*low DR* and *one material*) produce a better model when trained with the Mask R-CNN network. As soon as the training data gets more sophisticated, the trained YOLOv8 models show an increase in performance, while the Mask R-CNN models deteriorate. The best models for both algorithms are created training with a data set including real data. Here the Mask R-CNN performs slightly worse than the YOLOv8 for the purely real data set and significantly worse for the mixed data set. The performance of the Mask R-CNN model decreases if synthetic image data is added to a real image data training set.

4. DISCUSSION AND OUTLOOK

In this work we applied the Domain Randomisation method to close the reality gap between real and synthetic data. This was done by using our site creation tool, which creates synthetic rooms with multiple randomized

parameter, such as object material, environment setup and clutter objects. To evaluate the quality of different domain-randomized rooms in terms of their usefulness for algorithm training, we used the Truncated CLIP Entropy. With images from these rooms we trained two computer vision object detection algorithms to detect pipe segments within images. The produced models are evaluated using the Jaccard Score and the Mean Average Precision.

The TCE analysis tells us that by implementing more randomized rooms the complexity of the image data set increases. This effect gets weaker, with data sets already containing various environments. By inspecting the TCE results of the four levels of Domain Randomisation, it can be concluded that the variation of material has the highest influence (of our tested domain variables). The benefit of the additional clutter is also noticeable, however our method of varying the light elements did not influence the results significantly.

Our YOLO training showed that the synthetic data sets with lower TCE score also produce worse performing models. This is most likely due to the fact that the synthetic data does not exactly resemble our real data testing set. This could lead to the model learning features of pipes that are not present in the real environment. By randomizing the data, we try to randomized the learned features alongside, so that eventually the model is left with universal feature that apply to all pipe structures, synthetic and real. For the lower TCE valued data sets, the environments are not sufficiently chaotic, hence the model is not generalizing well enough. This also results in the fact, that for the *low DR* and *one material* data sets the increase of environments leads to a deterioration of the YOLOv8 model's prediction. The features are deeply manifested within the network and the models are over fitted to the *low DR* and *one material* data set.
By enlarging the number of rooms, we could increase the TCE value and the strongly correlating model performance. This is only possible if the randomization is strong enough, making the occurrence of similar environments seldom. The algorithm detects new sceneries and therefore improves its generalization ability, creating a model that predicts better on the test data set.

The best results are achieved by training with real-world images. A training set with only 279 images performed better than the best synthetic model with over 7000 images. However, adding synthetic data to the real data set creates a model, which performs better than the training of the two individual sets. If we increase the amount of added synthetic images, we improve our model's performance on average. Even with 0.97 % synthetic images, the 0.03 % real images make the model predict better than the pure synthetic data set. This suggests that the constructive aspects of the model features trained with real data are not equalized by the introduction of a vast number of synthetic images. Instead, the resulting model improves its generalization ability and therefore, performs even better.
However, this is only true if the generated synthetic data is randomized sufficiently. As the results for the different real and synthetic mix data sets suggest, only synthetic data with high TCE value (*no clutter* and *high DR*) improve the training with real-world training data.

As a potential risk we could identify the use of synthetic data with low TCE values. Mixing real data with synthetic data of low TCE values even reduces the model's performance.

Additionally, we found big differences between the two used algorithms. Mask R-CNN shows a much better performance with training data of low TCE values. Synthetic data of any kind, always led to a decrease in model performance for Mask R-CNN in contrast to YOLOv8.

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to Tamara Lenhard for her invaluable assistance with the use of the Colosseum tool. Her expertise and willingness to share her knowledge significantly enhanced our understanding and ability to use the tool effectively. This support was crucial for the successful completion of this project.

Internal Funding of DLR in Connection to Project AMG.

REFERENCES

[1] G. Jocher, A. Chaurasia, and J. Qiu, "Ultralytics YOLO," Jan. 2023.

[2] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick, "Mask r-cnn," *CoRR* **abs/1703.06870**, 2017.

[3] Y. Gong, G. Liu, Y. Xue, R. Li, and L. Meng, "A survey on dataset quality in machine learning," *Information & Software Technology* **162**, pp. 107268–107268, 2023.

[4] F. Poucin, A. Kraus, and M. Simon, "Boosting instance segmentation with synthetic data: A study to overcome the limits of real world data sets," in *2021 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, pp. 945–953, 2021.

[5] J. Tremblay, A. Prakash, D. Acuna, M. Brophy, V. Jampani, C. Anil, T. To, E. Cameracci, S. Boochoon, and S. Birchfield, "Training deep networks with synthetic data: Bridging the reality gap by domain randomization," 2018.

[6] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," 2017.

[7] L. da F. Costa, "Further generalizations of the jaccard index," *CoRR* **abs/2110.09619**, 2021.

[8] K. Wada, "Labelme: Image Polygonal Annotation with Python."

[9] M. A. . Research, "Codexlabsllc colosseum," 2022.

[10] T. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: common objects in context," *CoRR* **abs/1405.0312**, 2014.

[11] I. Matterport, "Mask_RCNN."

[12] F. Ibarrola and K. Grace, "Measuring diversity in co-creative image generation," 2024.