02456 DEEP LEARNING, DTU COMPUTE, FALL 2023

SEGMENTATION OF CAR PARTS IN COLLABORATION WITH DELOITTE CONSULTING

Sanem Leblebici (s222448) - Michał Lehwark (s222999) - Ari Menachem (s163956) - Elli Georgiou (s223408)

ABSTRACT

Image segmentation is a critical application in deep learning. In this project, the effectiveness of U-Net [1] and U-Net+ResNet [2] deep learning models were evaluated for car part segmenta- tion using a dataset provided by Deloitte Consulting. The dataset was enhanced through data reduction, data correction, and data augmentation methods. The project also highlights the important role of real-life images rather than using only synthetically generated data that does not fully resemble real life scenarios. The data augmentation techniques applied included image noise, color jitter, occlusions, gaussian blur, and horizontal and vertical flips. Over the training process, the impact of various degrees of data augmentation on the predicted segmentation accuracy was investigated for both deep learning models, as evaluated by mean dice coefficient score and mean pixel accuracy. The results proved a greater performance for the U-Net+ResNet model, achieving a mean dice coefficient score of 89.9% and mean pixel accuracy score of 90.1%. This project demonstrated the benefits of advanced feature extraction and data diversity in strengthening deep learning models for accurate car part segmentation.

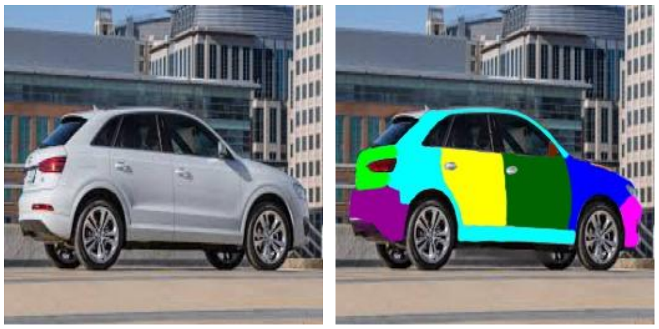
1. INTRODUCTION

In the rapidly evolving field of automobile technology, accurate segmentation of car parts becomes essential in many different kinds of applications including self-driving cars to damage assessment for insurance decisions. Considering the challenging nature of this task, selecting an effective deep learning model is important. This project aims to advance the state-of-the-art in car part segmentation by carefully comparing two neural network architectures: the established U-Net and the expanded U-Net+ResNet. It also highlights the importance of data preprocessing and explores different augmentation techniques. The initial dataset, provided by Deloitte, has been preprocessed to ensure high quality and equal distribution of data. This phase, which included data reduction and augmentation, helped to imitate the real-world situations that the model would face after deployment. Then, examining the impact of different augmentation techniques, both of the model architectures have been trained and evaluated, in order to determine an optimal balance of accuracy and efficiency. The findings of this report lay the ground for future development in achieving more precise segmentation models for car parts segmentation.

Code used to recreate results described below as well as the whole training pipeline can be accessed following the link: Github repository.

2. DATA AND PREPROCESSING

Deloitte Consulting provided the dataset used in this project. The dataset contained images of cars and their respective true segmentation masks. The dataset included about 3000 labelled synthetic car images (divided between 834 black car images and 2000 orange car images) and 168 labelled real car images. The segmentation masks included ten classes, corresponding to the background and the nine distinct car parts: Hood, front door, rear door, frame, rear quarter panel, trunk lid, fender, bumper, and rest of car. The car images were given as image files and the true segmentation mask labels were given as four-layer numpy arrays with the first three layers corresponding to the RGB-values of each pixel and the fourth layer containing the class label. In addition to the 'clean' car images, images with correctly placed segmentation masks were provided for visual purposes, as shown in the figure 1b.



(a) Example of real car photo (clean). (b) Example of real car photo (true segmentation mask included for visual purposes).

Fig. 1: Example of real car photo in the dataset.

The numpy array segmentation masks were of size 256x256 while the images were of various sizes. In order to prepare the data for training all images were downscaled to a size of 256x256 pixels. To avoid altering the images unnecessarily, the original aspect ratio of each image was kept constant. The size of 256x256 was then obtained by adding a padding of black color to the bottom of the downscaled image as done also by Deloitte in the numpy arrays.

Training the model (cf. section 4) with all labelled images excluding 30 real images set aside for the test set yielded unsatisfactory results, that were considered below expectations. Being unable to find a mistake in how the models' architecture was defined, the model was trained on a different image segmentation dataset obtained online. For this purpose, Aerial Semantic Segmentation Drone Dataset from kaggle has been obtained, which focuses on semantic understanding of urban scenes. Training with this data proved the model's ability to successfully perform image segmentation and achieve more satisfactory results. After training with a previously defined architecture and only for 5 epoche we were able to segment some parts of the scene as can be seen on figure 2, a performance previously unachievable when training with the initial dataset. Hence, the model's initially poor performance was attributed to shortcomings in the dataset. Several points of improvement of the dataset were identified, the details of which will be described in the following.

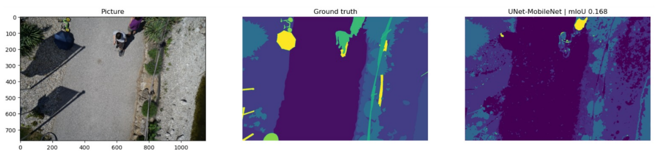


Fig. 2: Results of training UNET model on Aerial Semantic Segmentation Drone Dataset only after 5 epochs.

The synthetic car images that constituted the majority of the provided dataset included only two distinct car models, one black and one orange. Furthermore, the images were highly repetitive with many images being effectively identical. For this reason the synthetic images were severely reduced such that each image is clearly distinct from the others. Furthermore, several synthetic images that did not show the entire car were removed. An example is shown in figure 3. All together, the almost 3000 synthetic car images were reduced to 80 black car images and 80 orange car images. In the 168 real car photos several car models appear two or more times in the data, thus the number of car models the deep learning model is exposed to during training is significantly lower than the number of images in the dataset. Complex and varying shapes of car parts across different car models, constitutes a major limitation in the dataset. The real car photos vary in other aspects, such as angle, perspective, brightness, and hue and hence the real car photo data was not reduced.

Another issue with the provided dataset was that several of the numpy arrays given as true segmentation masks were noisy and incorrectly labelled, and did not match the segmented images that were provided for visual purposes. The noisy arrays were corrected by using pixel by pixel replacement based on the images with segmentation. This process is illustrated in figure 4.

Finally the size of the dataset was substantially increased using data augmentation methods. This is elaborated upon in section 3.

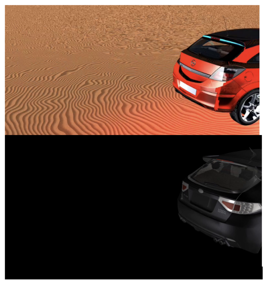


Fig. 3: Examples of excluded images.

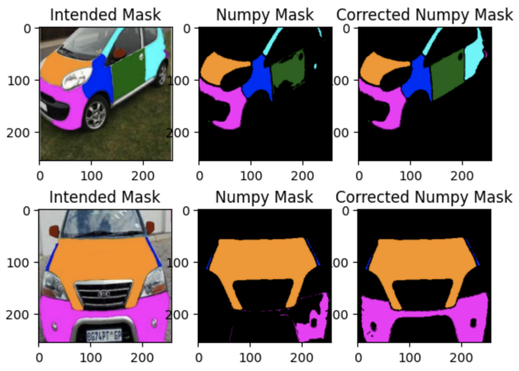


Fig. 4: Examples of corrected masks.

3. AUGMENTATION

A broad data augmentation process was applied in the effort to refine and increase the dataset for the car part segmentation task. By using this approach the variety and realism of the 138 training original real car photos were increased, which led to an improvement in the adaptability and accuracy of the implemented deep learning models.

Following Deloitte's requirements the test dataset should consist of a subset of the provided real car images (first 30 provided). Taking that into account the real car images were most valuable in the training process and preferably should be its majority to ensure the best performance. However, they contributed to only 4.5% of the whole data provided before any reductions. To mitigate this the augmentation process has been carried out only on that part of the dataset, leaving the synthetic images unchanged being a minority in the overall data. Each of the 138 original real car images was augmented in several ways, producing ten augmented copies of each image. This resulted in a total of 1,380 augmented training photos from the initial set of real images. The dataset preprocessing and augmentation pipeline is illustrated in figure 5.

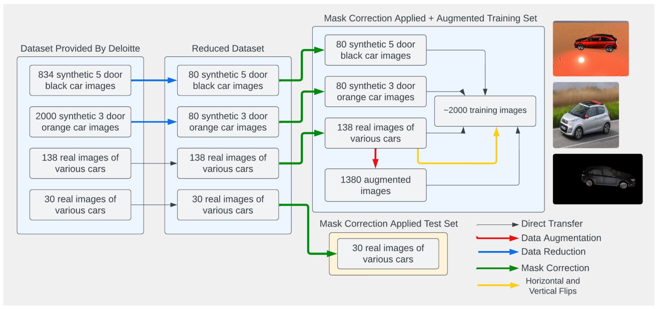

Dataset Provided By Deloitte
834 synthetic 5 door black car images
2000 synthetic 3 door orange car images
138 real images of various cars
30 real images of various cars
Reduced Dataset
80 synthetic 5 door black car images
80 synthetic 3 door orange car images
138 real images of various cars
30 real images of various cars
Mask Correction Applied + Augmented Training Set
80 synthetic 5 door black car images
80 synthetic 3 door orange car images
138 real images of various cars
1380 augmented images
~2000 training images
Mask Correction Applied Test Set
30 real images of various cars
Direct Transfer
Data Augmentation
Data Reduction
Mask Correction
Horizontal and Vertical Flips

Fig. 5: Dataset preprocessing and augmentation pipeline.

A number of augmentation strategies were chosen to expose models to a wide range of data imitating real-world scenarios and ensuring an improvement in the model's ability to generalize. Specifically the following augmentation techniques were applied in order to improve the dataset:

- **Image Noise.** Pixel noise was added to all photos, with the noise factor ranging within certain ranges (0.1-0.2, 0.1-0.11, 0.5-0.6, and so on). This technique aimed to simulate the effect of different environmental conditions on image quality.
- **Color Jitter.** Within preset boundaries, the brightness, contrast, saturation, and color of each image were randomly varied. This variation meant that the model could categorize car parts accurately under a variety of lighting and color situations.
- **Occlusions.** To simulate real-life visual obstructions, two 35x35 pixel black squares and their matching masks were superimposed on the photos at random locations. The likelihood of these occlusions happening was set to 0.3, adding complexity to the training data.
- **Gaussian Blur.** The photos were blurred with a Gaussian blur with a kernel size of 3. By this change we simulated the influence of atmospheric conditions or camera focus difficulties on acquired images.
- **Horizontal and vertical flips.** Horizontal and vertical flips were performed on each image in the dataset, thereby doubling the number of orientations in which the car parts may be viewed. This stage was important in teaching the model to recognize pieces despite their orientation.

Dataset	Image noise factor	Color jitter limit	Occlusion probability	Gaussian blur kernel size
0	0.1-0.2	0.2	0.3	3
1	0.1-0.11	0.2	0.3	3
2	0.5-0.6	0.2	0.3	3
3	0.1-0.2	0.05	0.3	3
4	0.1-0.2	0.5	0.3	3

Table 1: Augmentation parameters for the specific augmented datasets.

To verify the effectiveness of the applied data augmentation techniques a thorough review of the augmented photos and their related masks was conducted by picking a number of images randomly and examining them alongside their masks. This step ensured that the augmentation process did not corrupt or misrepresent the original data and that the model would be exposed to a wide range of different data simultaneously preserving the integrity and relevance of the training dataset.

For experimentation with differently augmented data, five different combinations of augmentation parameters where applied separately, creating five different datasets. In following sections it will be described how two different deep learning models, U-Net and U-Net+ResNet, are trained separately on each of the five augmented datasets. Table 1 shows the augmentation parameters used for each augmented dataset while figure 6 shows examples of augmented images.

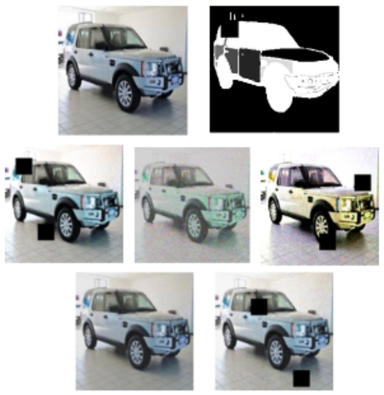


Fig. 6: Original image along with its corresponding mask and the different augmentation techniques.

4. MODEL

In order to tackle the challenge of image segmentation for car parts, two different models have been chosen for testing. U-Net has been chosen as a baseline model and later a more advanced architecture called U-Net+ResNet has been chosen to showcase potential differences in performance and improvements. They have been chosen based on their durability and track record of success in similar tasks. What is more, the latter model, with a slightly more advanced architecture incorporating pre-trained blocks was said to be able to pick up more details and features from the data earlier in the training process, which was worth exploring given the small areas of some car part classes take up in the images.

4.1. U-Net architecture

The initial choice of the baseline model was the U-Net, which is well-known for its performance in image segmentation tasks. The U-Net architecture was implemented in PyTorch by first creating a DoubleConv class, which represents a sequence of two convolutional layers, each followed by batch normalization and a ReLU activation function. This kind of structure is an essential part of the U-Net model, as it makes sure that it extracts the complicated characteristics from input photos. The U-Net model implemented included a contracting (downsampling) path, a bottleneck, and an expansive (upsampling) path. The downsampling was carried out using a series of DoubleConv layers and max pooling to reduce the spatial dimensions while improving the depth. The bottleneck, which was also built with the DoubleConv class, worked as a link between the downsampling and upsampling pathways. Transposed convolutions were used in order to improve the spatial dimensions in the upsampling section. Subsequently concatenation with the corresponding feature maps from the downsampling path were performed. This so-called skip connection process plays an essential role for the model to learn and maintain spatial information, which causes precise localization in the segmentation output. A diagram depicting the U-Net model is shown in figure 7.

4.2. U-Net+ResNet architecture

In order to improve the performance of the model even further, the ResNet architecture was combined with U-Net, yielding the U-Net+ResNet model. This model makes use of ResNet's complex feature extraction capabilities, particularly ResNet-50 block pre-trained on ImageNet, to extract more detailed and complicated characteristics from images. The U-Net+ResNet model has a similar structure to the regular U-Net model, but the beginning layers are replaced by ResNet-50 blocks. For the downsampling path, blocks from the pre-trained ResNet-50 model were used, causing more powerful feature extraction. The upsampling path was similar to that of the regular U-Net, with transposed convolutions for increased spatial dimensions and concatenation with ResNet-50 feature maps.

Both models' last layer was a convolutional layer that converted the feature maps into the number of output classes, which corresponded to the different car parts.

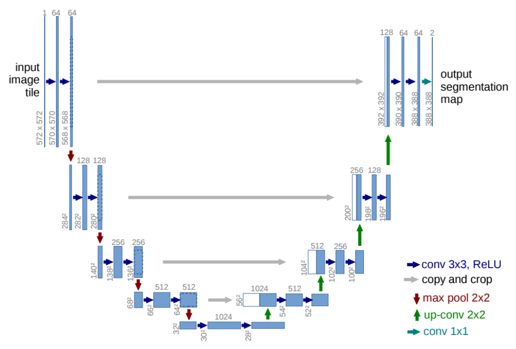


Fig. 7: U-Net model architecture [1].

5. TRAINING

5.1. Data and Image settings

To prepare for the training, the dataset was split into 85:15 training and validation sets. Thus, about 2,530 images were reserved for training and roughly 445 photos for validation. Each image was standardized to 256x256 pixels and normalized using ImageNet's mean and standard deviation values. However, as the normalization process didn't bring much value results as they stayed the same with or without it. Therefore, this technique has not been included in the final version of data preparation.

5.2. Training hyperparameters

The training parameters were carefully chosen in order to find the right balance between learning efficiency and stability. A batch size of 16 and a learning rate of 1e-3 were chosen. A dropout rate of 0.6 was included in order to reduce the possibility of overfitting, which is especially important in difficult segmentation tasks. To allow for enough time for learning and adaptation, the models were trained over 70 epochs. The U-Net and U-Net+ResNet models were initialized with 3 RGB image input channels and 10 class output channels. In the training loop, the focus was to maximize the model parameters and reduce the combined loss from cross-entropy loss and dice loss. The weighted cross-entropy loss was critical in correcting class imbalance by prioritizing minority classes and guaranteeing a more equal learning for all classes.

6. METRICS

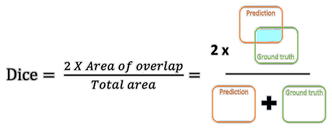
Two metrics were used to evaluate the performance of the U-Net and U-Net+ResNet models: The dice coefficient score and pixel accuracy, giving detailed insights into the models' segmentation accuracy and overall effectiveness.

6.1. Dice coefficient

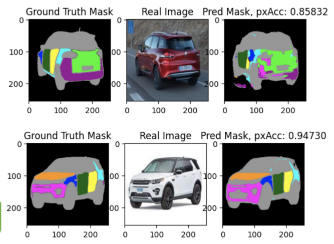
The dice coefficient score evaluates the overlap between the predicted segmentation mask and the ground truth segmentation mask. It is especially useful for determining how well the model captures the shape and area of the segmented car parts. It also takes the size of the split objects into account and penalizes false positives and false negatives. This coefficient was computed by determining the intersection of the predicted and true masks and adjusting it with a small epsilon to maintain numerical stability.

6.2. Pixel accuracy

Pixel accuracy measures the proportion of correctly predicted pixels out of the total number of pixels. It gives an overview of the model's performance throughout the full image. Furthermore, it is especially useful for determining how well the model works overall, although it can be misleading at times, especially when there is a class imbalance or the items of interest occupy a small section of the image. Figure 8 illustrates the concepts of dice coefficient and pixel accuracy.



(a) Dice coefficient.



(b) Mean pixel accuracy.

Fig. 8: Metrics used for performance assessment.

$$L_{CE} + L_{DL} = \left(-\sum_{i=1}^{n} t_i \cdot \log(p_i)\right) + \frac{1}{n}\sum_{i=1}^{n}\left(1 - \frac{2 \cdot \sum t_i \cdot p_i}{\sum t_i + \sum p_i + \epsilon}\right)$$

7. PERFORMANCE AND RESULTS

Results obtained from the experiments conducted with the U-Net and U-Net+ResNet models are outlined in figure 9. The results show that the application of data augmentation techniques was a determining factor in enhancing the model's adaptability and precision in segmenting car parts, as reflected by the dice coefficient score and pixel accuracy metrics. Data augmentation 0 dataset served as a control scenario. Training the baseline U-Net model with this data resulted in a dice coefficient score of 0.8844 and pixel accuracy of 0.8839.

Data augmentation 1 differed from the control scenario by being characterized by a narrower range of image noise factor (0.1-0.11 as opposed to 0.1-0.2 for the control scenario). Training the U-Net model with this data yielded a dice coefficient score of 0.8822 and a pixel accuracy of 0.8802, attesting to reduced precision as compared to the control scenario. This can be explained by the lower level of image noise applied during the augmentation, as the augmented images deviate less from the original images. Data augmentation 2 introduces a higher level of image noise being applied during the augmentation, the image noise factor ranging from 0.5-0.6. As expected, an increase in performance is observed relative to the previous experiment with a dice coefficient score of 0.8788 and a pixel accuracy of 0.8794. However, this does not explain why the results contrary to what would be expected are lower than the results for the control scenario, in which lower levels of image noise were applied. Data augmentation 3 maintains the same image noise factor range as the control scenario but has a reduced limit for the color jitter applied (0.05 as opposed to 0.2 for the control scenario). Hence, the augmented images deviate less from the original images than in the control scenario. Consistent with this the observed results indicate a reduction in performance with a dice coefficient score of 0.8709 and a pixel accuracy of 0.8728 when compared to the control scenario.

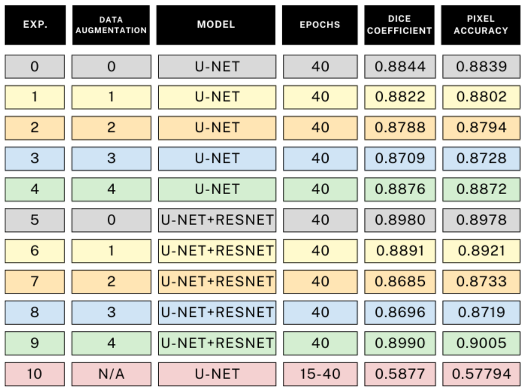
The greatest performance is observed for data augmentation 4. In this case the limit of the color jitter applied is increased substantially to 0.5. When training the U-Net with this data a dice coefficient score of 0.8876 and a pixel accuracy of 0.8872 is obtained. This can be explained by the fact that the augmented images differ more from the original images than in the previous experiments.

The expanded U-Net+ResNet model is trained with the same five different augmented datasets (experiments 5 through 9). While the results obtained indicate a similar pattern as described above a notable enhancement in performance was observed with the integration of ResNet blocks into the U-Net architecture. The hybrid model consistently outperformed the standard U-Net, with data augmentation 4 yielding the most robust results – a dice coefficient of 0.8990 and a pixel accuracy of 0.9005, underscoring the benefit of combining these architectures for complex image segmentation tasks.

However, an anomaly is observed with data augmentation 3, where the U-Net+ResNet model exhibited a decrease in performance, deviating from the expected trend. This encourages further investigation into the specific characteristics of the data augmentation techniques applied in this instance. Adjustments in the augmentation parameters or a deeper dive into the model's response to the techniques could uncover the causes of this discrepancy.

Finally an experiment was conducted with data that had been corrected (noisy and incorrectly labelled numpy segmentation masks, cf. section 2) but where data reduction and data augmentation had not been applied. Training the baseline U-Net model with this data for 15 epochs yielded a substantial reduction in performance with a dice coefficient score of 0.5877 and a pixel accuracy of 0.5779. Figure 10 and figure 11 illustrate the difference in performance with and without data augmentation, respectively. Overfitting occured beyond 15 epochs, but the model was trained for up to 40 epochs to ensure that performance would not increase.

The results indicate the superiority of the U-Net+ResNet model while the importance of appropriate data augmentation techniques and parameters is reinforced.



EXP.	DATA AUGMENTATION	MODEL	EPOCHS	DICE COEFFICIENT	PIXEL ACCURACY
0	0	U-NET	40	0.8844	0.8839
1	1	U-NET	40	0.8822	0.8802
2	2	U-NET	40	0.8788	0.8794
3	3	U-NET	40	0.8709	0.8728
4	4	U-NET	40	0.8876	0.8872
5	0	U-NET+RESNET	40	0.8980	0.8978
6	1	U-NET+RESNET	40	0.8891	0.8921
7	2	U-NET+RESNET	40	0.8685	0.8733
8	3	U-NET+RESNET	40	0.8696	0.8719
9	4	U-NET+RESNET	40	0.8990	0.9005
10	N/A	U-NET	15-40	0.5877	0.57794

Fig. 9: Experiment results.

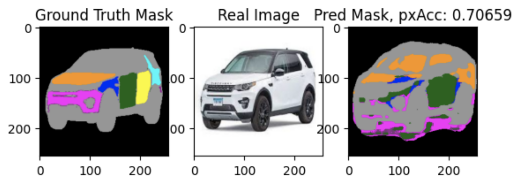


Fig. 10: Prediction without applying data augmentations.

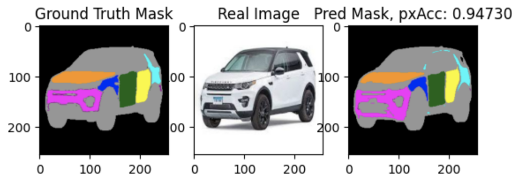


Fig. 11: Prediction after applying data augmentations.

8. CONCLUSIONS AND FUTURE WORK

After improving the segmentation masks and increasing the dataset size using data augmentation, the trained model was able to effectively predict segmentation masks of the images in the test set. Accuracies achieved ranged from 86.9% to 89.9% for dice score and from 87.2% to 90.1% for pixel accuracy score. The best results for both dice score and pixel accuracy were achieved using the U-Net + ResNet model with data augmentation 4 as shown in figure 9. This improvement is partially attributed to the increased color jitter used during augmentation, indicating its effectiveness in improving model precision. However, results revealed that establishing the precise influence of specific augmentation approaches on model performance was challenging. The similarities in results across multiple augmentation settings highlight the potential advantages of investigating other augmentation procedures for more definitive findings.

It was deemed that the main limitaton for predicting segmentations masks accurately was related to the dataset's diversity and to obtain both more and higher-quality data. With regard to the need for more data: Some characteristics of the segmentation masks vary from car model to car model. For example on the leftmost image in figure 11 it is clear to see that the lower part of the car's front bumper is not included in the true segmentation mask of this particular car model. However, for many car models this part is included. Thus as it is shown in the rightmost image of figure 11 this part is predicted as a part of the front bumper. While this prediction applies to the vast majority of car models, it does not apply to this particular one. Therefore, in order for the model to accurately differentiate such fine-grained characteristics, it must be trained on a wide range of car models, and this cannot be accomplished just by augmenting the same limited set of car models with varied image modifications. As for the higher quality data, it is obvious that the dataset contains inconsistencies. Some segmentation masks include the bottom areas of the car's sides within the frame (marked by turquoise blue), while others do not. As a result, while the model provides correct predictions, the measured accuracy is compromised. To enhance prediction quality, it is imperative to establish consistency within the segmentation masks. This method, however, may be time-consuming because it may involve manual changes to the segmentation masks.

8.1. Final remarks on improving the 90% accuracy

As outlined in the conclusions, data quality had immense influence on the accuracy of the results. The overall results did not exceed around 90% in the final version of the project. Therefore, it is felt that the topic of hyper-parameters should be addressed and whether changing them could have improved the results. Once the best performing model was found as outlined in Section 7, several different versions of hyper-parameters were experimented with. Possible combinations of 3 different learning rates, 3 different batch sizes and with and without dropout were tested. No significant improvement on the accuracy was observed with these experiments and therefore the hyper-parameters mentioned in Section 5 were used in the final version of the project.

References

[1] O. Ronneberger, P. Fischer, and T. Brox, *U-net: Convolutional networks for biomedical image segmentation*, 2015. arXiv: `1505.04597 [cs.CV]`.

[2] X. Chen, L. Yao, and Y. Zhang, *Residual attention u-net for automated multi-class segmentation of covid-19 chest ct images*, 2020. arXiv: `2004.05645 [eess.IV]`.