Not Using the Car to See the Sidewalk — Quantifying and Controlling the Effects of Context in Classification and Segmentation Supplementary Material

Rakshith Shetty[1] Bernt Schiele[1] Mario Fritz[2]

[1]Max Planck Institute for Informatics, Saarland Informatics Campus
[2]CISPA Helmholtz Center for Information Security

[1]firstname.lastname@mpi-inf.mpg.de [2]firstname.lastname@cispa.saarland

In this supplementary material we present the following additional details to support the results in the main paper. Section 1 provides the details about the in-painting network architecture. Section 2.1 provides a discussion on the co-occurrence statistics on COCO which influences the classifier behavior and provides a visualization of robustness compared to performance. Section 2.2 provides visualization of robustness statistics of the segmentation model and provides ablations to test the importance of removal and in-painting in data augmentation. Additional qualitative examples are provided in Figures 3 and 6.

1. Object removal model

To remove objects we use the ground truth segmentation masks and dilate them by a small factor (5 in the coco dataset and 7 in the ADE20k dataset). This dilated mask is multiplied with the input image to remove the target object from the image. Then the masked image and the mask is passed to an in-painting network which fills the masked area with a plausible background texture. We use the in-painting architecture and the training procedure proposed in [1]. Table 1 and 2 present the detailed architecture of the in-painting network. We will make the code and pre-trained in-painter models available after the review process.

Masked Image + mask
Conv 4x4, 64 filters, stride 1
Conv 4x4, 128 filters, stride 2
Conv 4x4, 256 filters, stride 2
Conv 4x4, 512 filters, stride 2
Residual Block, 256 filters
Residual Block, 256 filters
Residual Block, 256 filters
Residual Block, 256 filters
Residual Block, 256 filters
Residual Block, 256 filters
Upsample + Conv 3x3, 256 filters
Upsample + Conv 3x3, 128 filters
Upsample + Conv 3x3, 64 filters
Conv 7x7, 3 filters, stride 1

Table 1: In-painting model architecture starting with input in the first row to the output layer in the last. Each convolutional layer is followed by a Instance Norm layer and a Leaky Relu non-linearity with slope 0.1

Conv 3x3, n filters, stride 1
Instance Norm
Leaky Relu (slope 0.1)
Conv 3x3, n filters, stride 1
Instance Norm
Leaky Relu (slope 0.1)

Table 2: Architecture of the residual block with n filters

2. Analyzing robustness to context

In the main paper, we presented our analysis showing that the classification and segmentation models are sensitive to context and their predictions are significantly affected when presented with edited images with context objects removed. In the following sub-sections we present additional visualizations to support these arguments.

2.1. Image-level Classification

Co-occurrence of objects. An important factor which causes the image-level classification models to use contextual dependencies is the co-occurrence distribution of objects.Many objects in COCO have a strong co-occurrence relation with other objects. We quantify this using the normalized co-occurrence counts for each object with others given by

$$NC(c_i, c_j) = \frac{\text{Count}(c_i \cap c_j)}{\text{Count}(c_i)}$$

. This matrix is visualized in Figure 1. $NC(c_i, c_j)$ takes value between 0 and 1 and represents the fraction of images containing object c_i, which also contains object c_j. We can

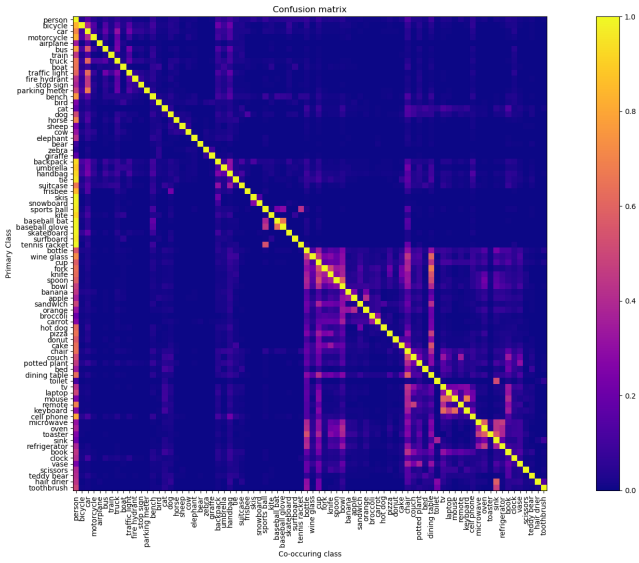

Figure 1: Co-occurrence ratio, $N(c_i, c_j)$ of objects on the COCO dataset

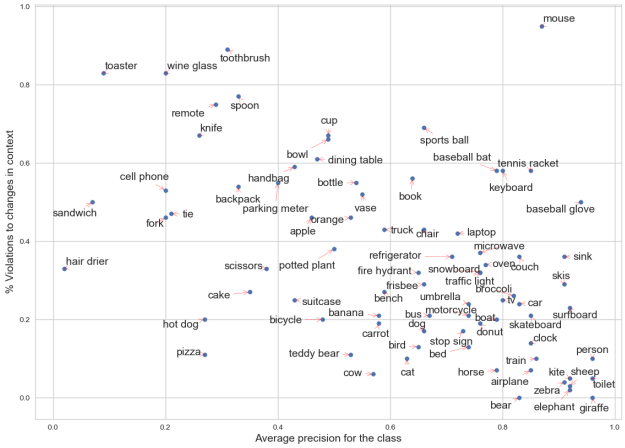

Figure 2: Comparing class-wise average precision to the % of violations in changes to context. Many well-performing categories (high mAP), have high percentage of violations, including mouse, tennis racket, keyboard, book, and sink.

see that, for classes like skateboard, surfboard, tennis racket and handbag, this ratio is very high ($\geq 90\%$) with the person class, since these classes often occur with a person holding or riding them. We also see that the matrix is not symmetric. This is because, while the skateboard might occur always with a person, but person class occurs in various contexts without skateboard. However, for some groups of objects like mouse, keyboard and monitor, and spoon, fork, and cup have symmetric co-occurrence relationship.

For many categories, including the cases discussed above, the co-occurrence ratio is very high ($> 60\%$). This causes problems for object classifiers of these categories, as we see in the analysis presented in the main paper. When a small or difficult to detect object class like mouse or skateboard, frequently co-occurs with a more easy to detect object class like monitor or person, the classifiers tend to overuse the contextual relationship for making their classification decisions instead of visual evidence for the object of interest. This leads to failures when the context is different or the object occurs without context.

Relation of performance to robustness. As discussed in section 4.1.2 in the main paper, we find that many well-performing object classes in terms of average precision (AP) perform poorly in terms of robustness. To show this we plot the per-class average precision against the worst-case robustness metric $V^{\min}(c_i)$ in Figure 2. We can see for example that classes like mouse, tennis racket, sports ball, baseball bat and book which have high AP (geq 0.6) have poor robustness ($V^{\min}(c_i)$ ¿ 0.5). In all these cases, the classifier seems to predominantly use contextual objects to make their predictions and achieve high average precision. But they fail when presented with object-without-context and context-without-object images, usually scoring the context-without-object images higher. This is also seen in further visual examples presented in Figure 3. Interestingly visually distinct classes like zebra, elephant, giraffe achieve high AP, while also being robust as seen in Figure 2 .

2.2. Semantic Segmentation

Visualizing robustness metrics. We compute the robustness metric $AR(c_i, c_j)$, which measures the ratio of instances when segmentation of class c_i is affected by the removal of class c_j, in the ADE20k dataset for the Upernet [2] model. This is visualized in Figure 4. The y-axis the affected class and the x-axis is the removed object class. We show the rows and the columns which have atleast one entry > 0.1, for readability. We can see that the $AR(c_i, c_j)$ matrix is very sparse, indicating that the segmentation is not affected by all removal, but of only specific classes. As discussed in section 4.2.2 of the main paper, we can see that classes like road and sidewalk depend on the class car. The sidewalk is also to an extent affected by removal of trees.

To measure the direction of the effect, that is if removal of context harms or improves the segmentation of a class, we visualize the average change in IoU in Figure 5. Surprisingly, we find that not all context removal negatively affects the segmentation. Sometimes removing an object helps the model to resolve ambiguities and fix the segmentation of other objects. We can see in Figure 5 that while majority of change is negative, for a few pairs of objects removal positively affects the IoU. For example removing *lamp* class improves the segmentation of ceiling *light*. Similarly, removing *armchair* improves segmentation of *chair*

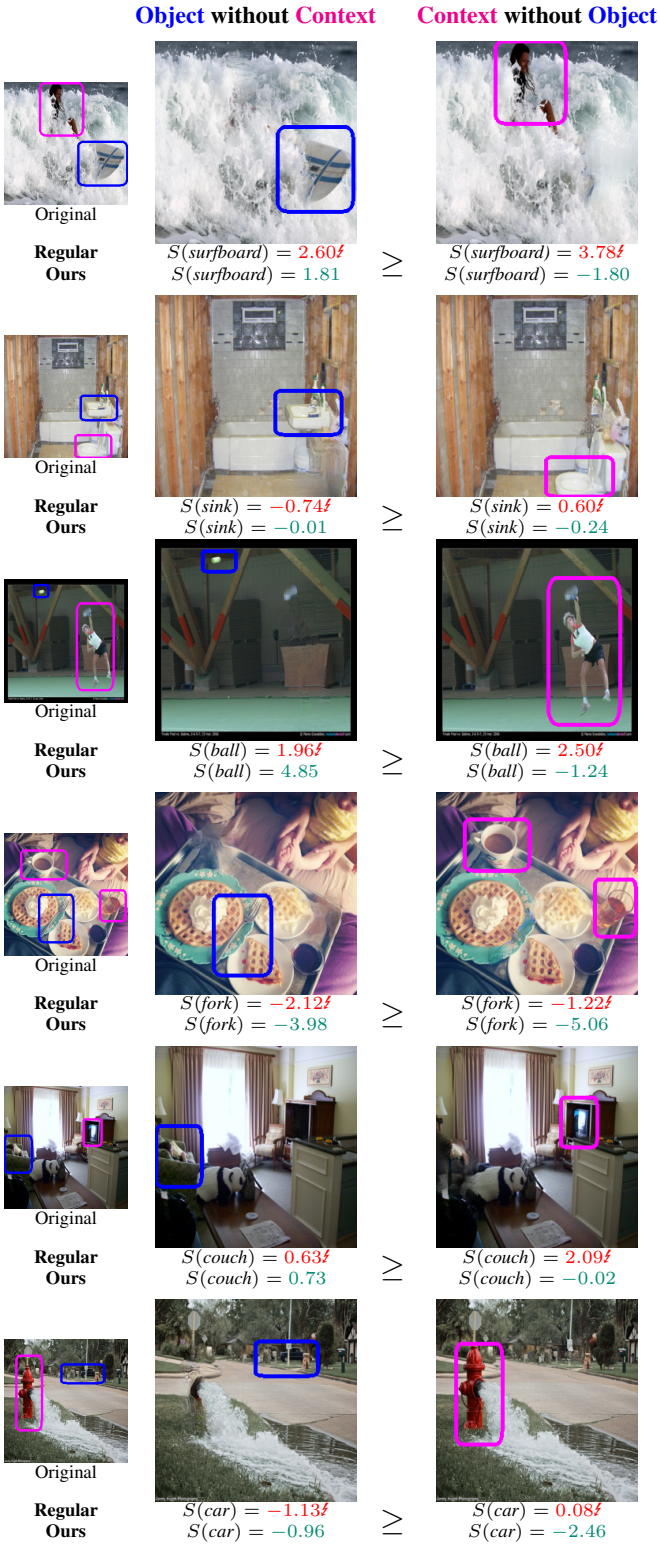

Figure 3: Context violations by image-level classifier. The primary object is marked with blue box and the context object is marked with magenta. The first column shows the original image, middle shows the image with only object and the third with only the context. We see that the baseline classifier depends heavily on the context and always scores the context only images (last column) higher than the image with only the primary object (middle column). The data augmented model does better and gets the ordering right.

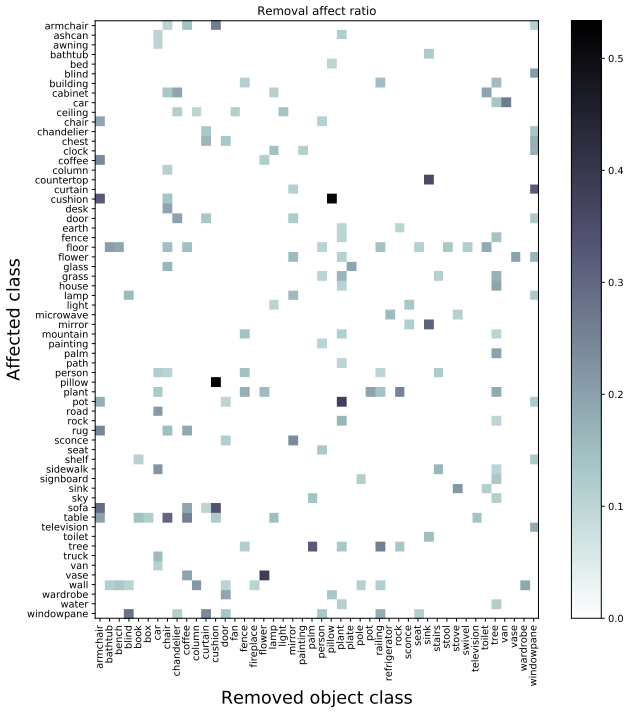

Figure 4: Visualizing frequency with which classes are affected by removal of other objects. Y-axis are the affected objects and the x-axis shows the removed objects

and *sofa* classes, since the ambiguity is resolved.

Ablation on data augmentation. To understand if removal and in-painting is really needed for data augmentation, we conduct two ablation studies which are presented in Table 4. First we train a version of the baseline model where for each training sample we randomly select an object and set its label to 'ignore'. We use the same sampling strategy as *sizebased* data augmentation model. Hence this mimics exactly the training procedure in DA (sizebased), except without actually removing the object. Comparing the results of this model (No removal (sizebased)) to the data augmented version, we see that removing the object is necessary and simply ignoring the label leads to a severe drop in performance (0.354 vs 0.379). Similarly, in the second experiment we train a model with the object removed but without in-painting. In this case we can see from Table 4 that, having in-painter during augmentation is slightly better than the model without (0.379 vs 0.375).

Further visual examples of the sensitivity of the baseline Upernet [2] model to contextual changes and the robustness provided by data-augmentation is seen in Figure 6.

Confirming the source of sensitivity. Finally we conduct an additional experiment to verify that the volatility we see in the output of the segmentation models on edited images are due to removal of context objects and not due to edit-

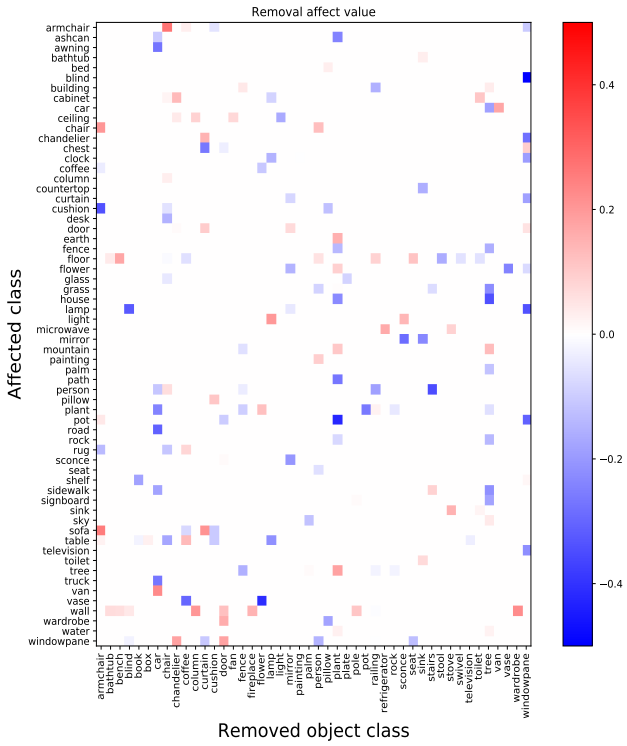

Figure 5: Visualizing the mean change in the IoU of object segmentation with context object removal. Y-axis are the affected objects and the x-axis shows the removed objects

ing artifacts. To test this we observe the predictions of the model on three version of the input image. First is the original image. Second is the image with a context object removed. Finally the last image is the false edited image created by masking and in-painting the input image with the same mask as the context object, except with a horizontal flip. Thus the context object is not removed in the false edited image, but a similar shape and size region is removed and in-painted in a different part of the input image. All three images are fed to the segmentation models and the output is shown in Figure 7. We can see that the Upernet model output is virtually identical on the original image and the false edited image(third row). However the segmentation on the edited image with context object removed is significantly different (second row). This indicates that the segmentation models are not affected by the editing artifacts but by the removal of the context objects as claimed in the paper.

Context sensitivity of different architectures.. In order to understand if the context sensitivity we reported extends to other network architectures, we tested additional models on the segmentation task. Table 3 presents context sensitivity of the sidewalk class to removal of cars, for four new architectures (pretrained models obtained from [2]). Despite using different encoders and decoders and having very different mIoU, all the four models exhibit similar sensitivity of sidewalk class to removal of car (18-22%) as reported for our original baseline model (*resnet-50). The overall sensitivity matrix $AR(c_i, c_j)$ looks similar to the one shown in Figure 4. We also see from Table 3 that data augmented model still achieves the lowest sensitivity (14%). The above experiment confirm that context sensitivity is not specific to a network architecture, but is seen across different models.

Encoder	Decoder	mIoU	Sensitivity of sidewalk to car
mobilenet	conv module [2]	0.324	18%
resnet-18	ppm [3]	0.380	18%
resnet-50	ppm [3]	0.408	20%
resnet-101	upernet [2]	0.420	22%
* resnet-50	upernet [2]	0.377	22%
* resnet-50 + DA	upernet [2]	0.385	**14%**

Table 3: Context sensitivity of different networks on ADE20k. Models marked with * are also reported in the main paper and are trained by us with batch size of 6, due to limited GPU memory. All the other models are trained by [2] with batch size of 16

Model	Removed Pixels	ADE20k mIoU	ADE20k Acc
Upernet[2]	-	0.377	78.31
No removal (sizebased)	Ignore	0.354	77.45
No inpainter (sizebased)	Ignore	0.375	78.25
DA (sizebased)	Ignore	0.379	78.31

Table 4: Data augmentation results on ADE20k dataset

References

[1] R. Shetty, M. Fritz, and B. Schiele. Adversarial scene editing: Automatic object removal from weak supervision. In *Advances in Neural Information Processing Systems (NIPS)*, 2018. 1

[2] T. Xiao, Y. Liu, B. Zhou, Y. Jiang, and J. Sun. Unified perceptual parsing for scene understanding. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018. 2, 3, 4

[3] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid scene parsing network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 4

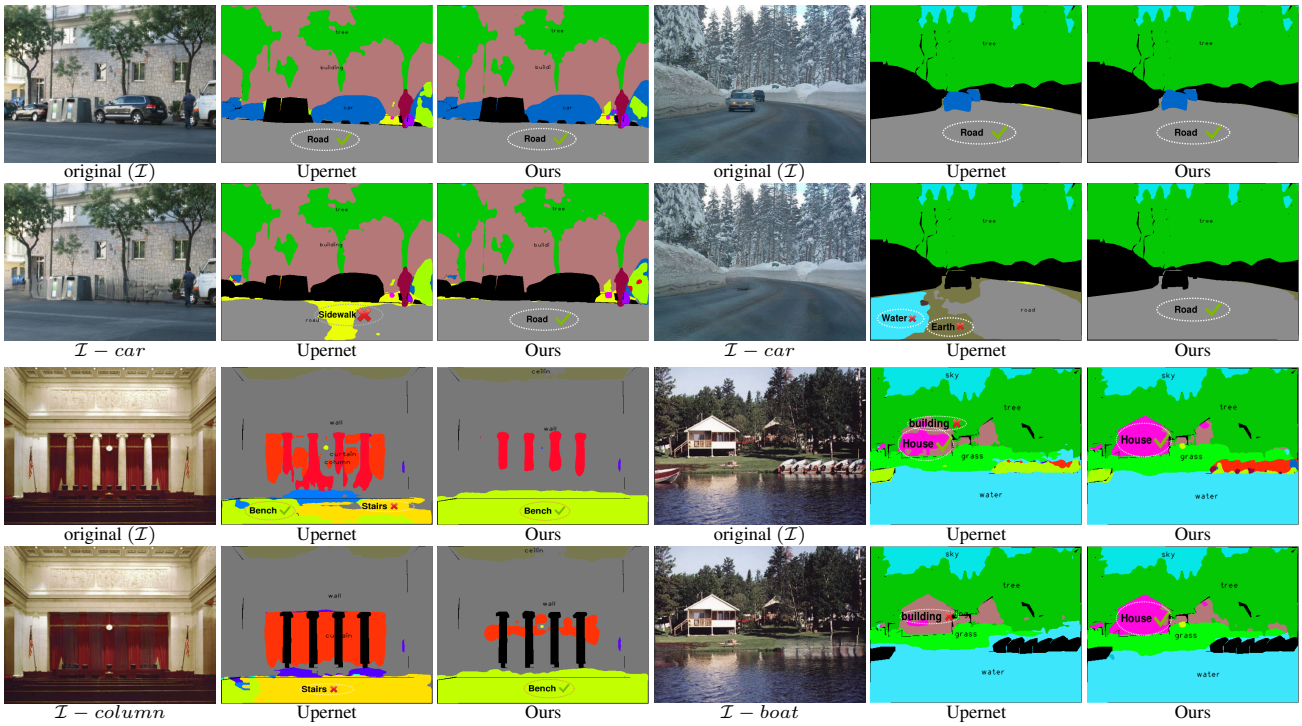

Figure 6: Examples of segmentation failures due to removal of a single context object. We see the segmentation of road, bench and house affected significantly when context objects like car, columns and boat is removed (comparing odd and even rows). Model trained with proposed data-augmentation is more robust to these changes.

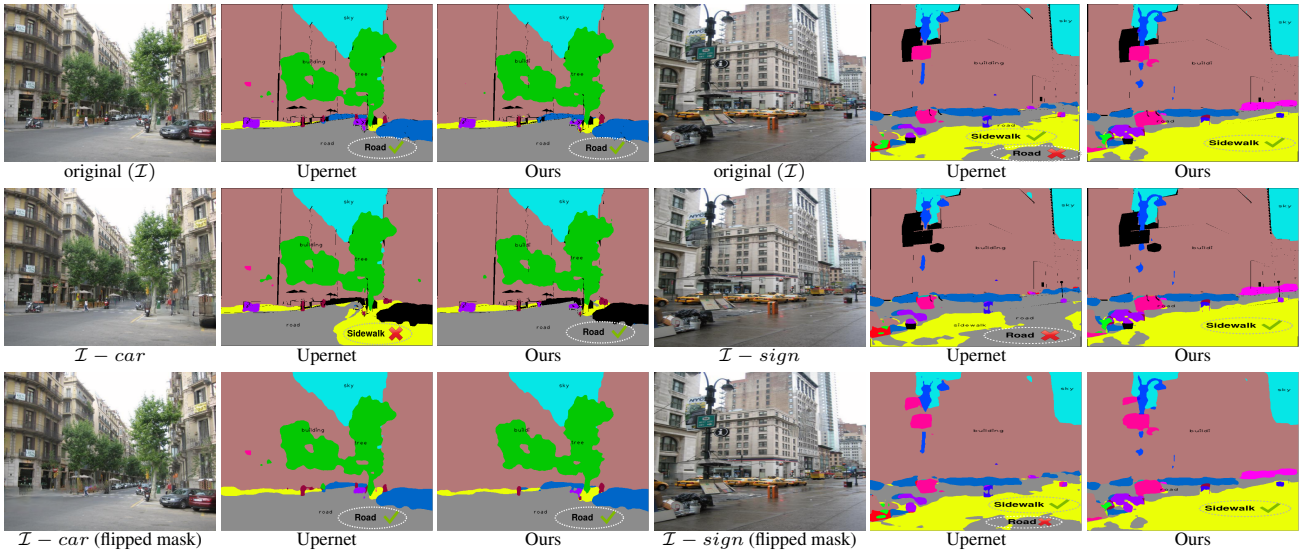

Figure 7: Experiment to verify that the volatility of the segmentation output is due to object removal and not due to editing artifacts. First row shows original images and the segmentation produced for them. Second row shows the edited images with an object removed and the segmentation output for them. Here we can see the segmentation output of Upernet significantly affected by the removal of car and sign. Final, row shows the original image edited with the same object mask as the second row, but horizontally flipped. Thus the object is not removed, but a different part of the image is edited with the same mask. We can see here that the segmentation is not affected at all by this edit and is very similar to the segmentation produced by the original image.