A Reproducibility

A.1 Training Protocol

For the experiments we follow the same train, test and validation splits as [3]. Instead of using ImageNet pre-trained models, we use the same randomly initialized ERFNet in all our experiments. For optimization we utilize SGD with wight decay factor of 3×10^{-4}, momentum of 0.9 and an initial learning rate of 0.07 for the first task and 5×10^{-4} for the second task. The learning rate is divided by 2 if the validation loss is not reducing for 8 consecutive epochs. We train the model in each task for 100 epochs with a batch size of 16. After training on the entire task sequence the model is evaluated on the validation set of PascalVoc2012. During training we use the same augmentations as [3], but use the implementations of Albumentations.

A.2 Continual Hyperparameter Selection

To select the hyperparameters for the continual learning methods we follow the Continual Hyperparameter Framework of [7], by firstly choosing the appropriate learning rate that achieves the highest accuracy on the new task and consecutively tuning the specific continual learning hyperparameters. This lead to the following hyperparameters for the respective methods:

- Fine-Tuning: $\text{lr}_0 = 0.07, \quad \text{lr}_1 = 5 \times 10^{-4}$
- EWC[6]: $\lambda = 10000$
- MAS: $\lambda = 5000$
- LwF: $\lambda = 6, \quad T = 2$
- MiB: $\lambda = 25, \quad \alpha = 1$

For all methods we use the same $\text{lr}_0 = 0.07$ and $\text{lr}_1 = 5 \times 10^{-4}$. In the experiments with UNCE and Weight Normalization we use the exact same parameters.

A.3 Implementation:

All of our experiments are conducted using PyTorch in combination with Pytorch Lightning. We use the original PyTorch implementation of ERFNet provided by [36] which can be found at: https://github.com/Eromera/erfnet. The implementation of the unbiased cross-entropy loss is taken from [3], which can be found at: https://github.com/fcdl94/MiB. For the continual learning algorithms we adapt the implementation of [25] for the use in semantic segmentation. The reference code can be found at: https://github.com/mmasana/FACIL.

[6] In order to use such a high value for λ we clip the gradient norm at a value of 10 to avoid exploding gradients.

A.4 Decoder Retraining

For Decoder Retraining we freeze all layers of the encoder of f_1 and set them in evaluation mode. We train the decoder with SGD with weight decay factor of 3×10^{-4}, momentum of 0.9. We use an initial learning rate of 0.07, but replace the learning rate schedule of the previous experiments with a cosine annealing learning rate schedule. The model is trained for 30 epochs on the PascalVoc-2012.

A.5 Training Split Sizes

The number of images in each training subset are: *Overlapped* - [T1: 9568, T2: 2145], *Disjoint* - [T1: 8437, T2: 2145], *Full-Disjoint* - [T1: 8437, T2: 1014].

B Results for different CNN-Architectures

We validated our findings on the widely adopted U-Net [37] and DeepLabv3+ [4] each with ResNet101 Backbones, compare Table 5. When using the cross-entropy loss (CE) the results of the different architectures are similar, because the information of old classes is completely overwritten, as the models are optimized to assign the background class to old classes. However, when using UNCE we see that the models with higher learning capacity can retain much more of the information of old classes than the smaller ERFNet, leading to much better results for DeepLabv3+. In the *full-disjoint* setting the use of the UNCE loss does not lead to significant improvements even for DeepLabv3+ and U-Net. This further strengthens our observations of the role of the background class. When looking at the internal representation shifts in Fig. 7, DeepLabv3+ behaves similar to ERFNet, as it retains high similarity throughout the network and suffers from a sudden drop in similarity in the layers of the decoder. On the other hand, for U-Net even early layers (starting from layer 10) are affected by a representation shift. A likely explanation are the skip-connections that connect early layers directly to decoder layers in the network. This also shows that the architecture choice has a significant impact on the internal representation drift. Future research directions could investigate how the recent trends in neural architectures impact the internal representation drift during continual learning, specifically recently proposed normalization layers and vision transformer are of high interest.

Table 5: Results of Semantic Segmentation on Pascal-VOC 2012 in Mean IoU (%) on the disjoint and full disjoint settings.

	PascalVoc 15-5 - (disjoint)						**PascalVoc 15-5 - (full-disjoint)**					
	CE			**UNCE**			**CE**			**UNCE**		
Method	*0-15*	*16-20*	*all*	*0-15*	*16-20*	*all*	*0-15*	*16-20*	*all*	*0-15*	*16-20*	*all*
ERFNet [36]	4.6	23.0	9.0	10.4	21.8	13.1	5.1	16.3	7.8	6.0	15.5	8.3
DeepLabV3+ [4]	4.6	24.7	9.4	21.4	23.7	21.9	5.7	16.8	7.9	5.7	16.2	8.2
UNet[37]	4.6	25.6	9.6	12.7	25.3	15.7	5.3	16.2	7.9	6.1	16.6	8.6

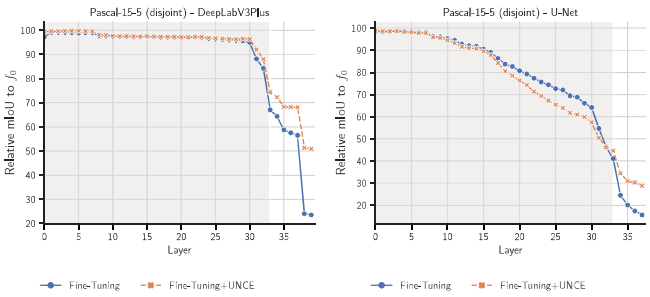


Fig. 7: Activation drift between f_1 to f_0 measured with Frankenstein Networks for DeepLabv3+ [4] (left) and U-Net [37] (right).

C Dr. Frankenstein with and without Stitching Layer

We also compare the results of layer stitching with and without the additional 1×1 convolutional layer in Fig. 8. The stitching layer is initialized with least-squares-matching and optimized with cross-entropy loss, as described in [5]. The most notable difference between two Dr. Frankenstein variants is the initial drop in similarity in the early layers that we observe only with the additional stitching layer. This was also reported in the experiments by the authors of Dr. Frankenstein, but as of now there is no explanation why no high accuracy stitching can be learned for those layers [5]. However, with the exception of the unexpected high dissimilarity in early layers, both variants show the same trends: high similarity in the encoder; sudden drop of similarity from layer 17 to 18 for MAS, EWC and Fine-Tuning and highest similarity at final layer by MAS-UNCE and LwF. However, we also note that from layer 4-15 the addition of the stitching layer leads to higher similarity, while for the decoder layers we observe lower similarity compared to Dr. Frankenstein without the stitching layer.

D Dr. Frankenstein vs. Centered Kernel Alignment

Centered Kernel Alignment (CKA) [20] is a similarity index that measures the similarity between internal representation of neural networks. In continual learning CKA has recently been used to measure the activation drift of the intermediate layers of a neural network [33]. Csiszárik *et al.* [5] investigated the relationship between representational similarity that is measured by CKA and functional similarity measured by Dr. Frankenstein. They demonstrate that a high CKA score between activations does not infer that the networks have functional similarity. Meaning that high functional similarity can be retained while simultaneously the representational similarity measured by CKA decreases. In Fig. 9 we compare the functional similarity measured by Dr. Frankenstein and the representational similarity measured by CKA. In line with previous assumptions the CKA analysis also confirm that the representations of the first 4 layers

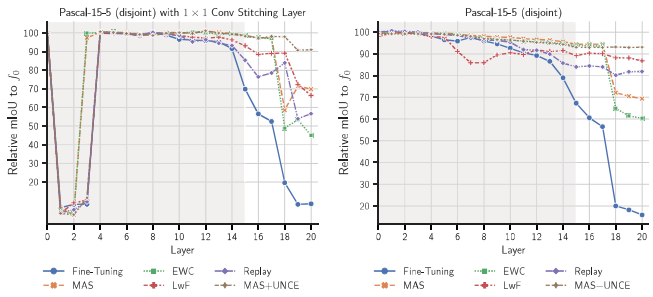


Fig. 8: Activation drift between f_1 to f_0 measured by Dr. Frankenstein with an additional 1x1 convolutional stitching layer (left) [5] and w/o the stitching layer, as used in the paper (right).

are not affected by representational drift. However, in the decoder we observe that the representational drift between f_0 and f_1 for EWC and MAS is not as pronounced as for the measured by Dr. Frankenstein. This indicates that the activations only undergo a minor change in terms of representational distance, but that these minor activation changes have a severe impact on the functional similarity of the model.

E Visualization of the Bias Values of the Classification Layer

The bias towards most recent classes and the background manifests itself in the bias values of the final convolution layer. The visualization in Fig. 10 demonstrates that all methods have obvious increased bias values for the background class and classes of T_1, with the exception of LwF in the *Disjoint* setting. However, once old classes disappear from the background of the images in the *Full Disjoint* setting LwF shows similarly high bias values for new classes as well.

F Confusion Matrices PascalVoc-15-5

In the following we show all the confusion matrices for the *Overlapped*, *Disjoint* and *Full-Disjoint* setting. The confusion matrices clearly illustrate that in the *Full-Disjoint* the background bias is much less pronounced whereas the class confusion is more severe.

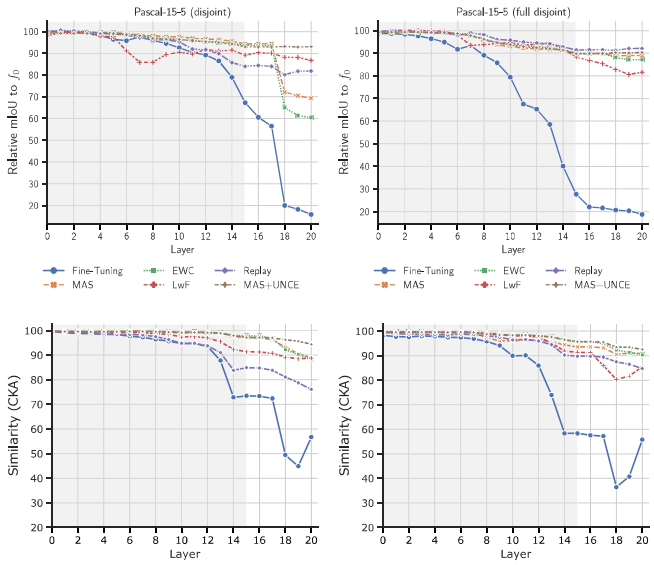


Fig. 9: Activation drift between f_1 to f_0 measured on the first task data by Frankenstein Networks (top row) and CKA (bottom row).

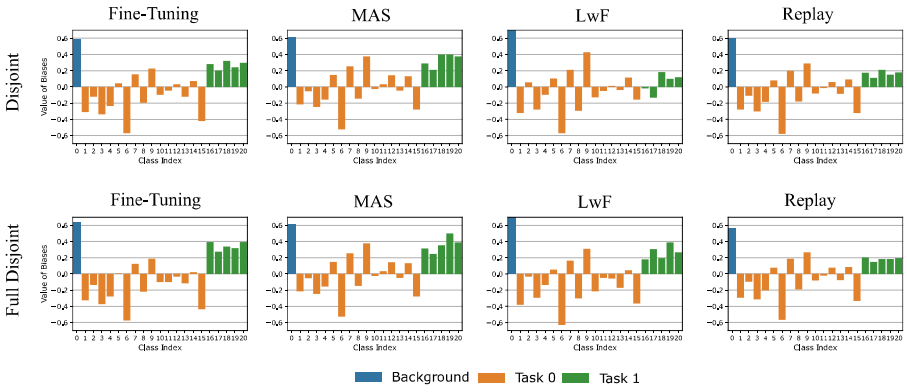
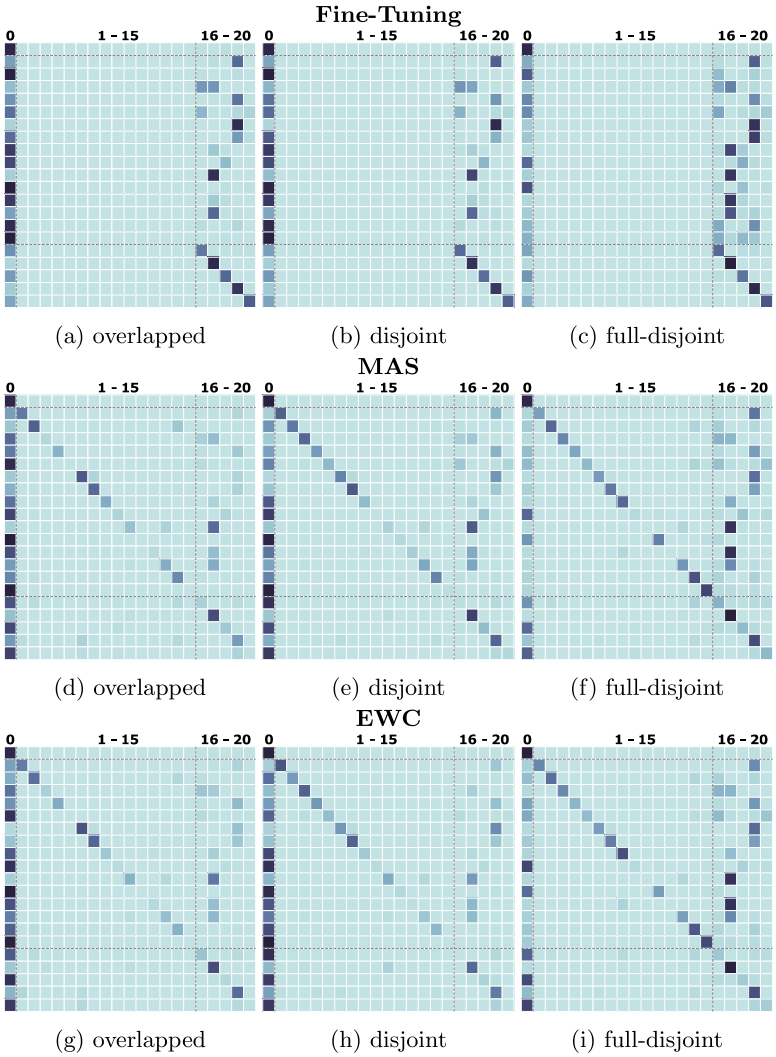


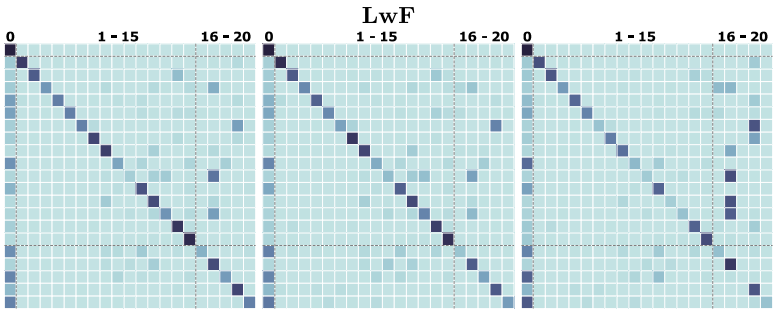
Fig. 10: Bias Values of the final convolutional layer after training on the *Disjoint* and *Full Disjoint* PascalVoc-15-5 split.



(a) overlapped (b) disjoint (c) full-disjoint

(d) overlapped (e) disjoint (f) full-disjoint

(g) overlapped (h) disjoint (i) full-disjoint



(a) overlapped (b) disjoint (c) full-disjoint

(d) overlapped (e) disjoint (f) full-disjoint