Principles of Forgetting in Domain-Incremental Semantic Segmentation in Adverse Weather Conditions

Tobias Kalb  
Porsche Engineering Group GmbH  
Weissach, Germany  
tobias.kalb@porsche-engineering.de

Jürgen Beyerer  
Fraunhofer IOSB & Karlsruhe Institute of Technology  
Karlsruhe, Germany  
juergen.beyerer@iosb.fraunhofer.de

Abstract

Deep neural networks for scene perception in automated vehicles achieve excellent results for the domains they were trained on. However, in real-world conditions, the domain of operation and its underlying data distribution are subject to change. Adverse weather conditions, in particular, can significantly decrease model performance when such data are not available during training. Additionally, when a model is incrementally adapted to a new domain, it suffers from catastrophic forgetting, causing a significant drop in performance on previously observed domains. Despite recent progress in reducing catastrophic forgetting, its causes and effects remain obscure. Therefore, we study how the representations of semantic segmentation models are affected during domain-incremental learning in adverse weather conditions. Our experiments and representational analyses indicate that catastrophic forgetting is primarily caused by changes to low-level features in domain-incremental learning and that learning more general features on the source domain using pre-training and image augmentations leads to efficient feature reuse in subsequent tasks, which drastically reduces catastrophic forgetting. These findings highlight the importance of methods that facilitate generalized features for effective continual learning algorithms.

1. Introduction

Semantic segmentation is widely used for environment perception in automated driving, where it aims at recognizing and comprehending images at the pixel level. One fundamental constraint of the traditional deep learning-based semantic segmentation models is that they are often only trained and evaluated on data collected mostly in clear weather conditions and that they assume that the domain of the training data matches the domain they operate in. However, in the real world, those autonomous driving systems are faced with constantly changing driving environments and variable input data distributions. Specifically, changing weather conditions can have adverse effects on the performance of segmentation models.

Therefore, a semantic segmentation model needs to be adapted to these conditions. A naive solution to this problem would be to incrementally fine-tune the model to new domains with labeled data. However, fine-tuning a neural network to a novel domain will, in most cases, lead to a severe performance drop in previously observed domains. This phenomenon is usually referred to as catastrophic forgetting and is a fundamental challenge when training a neural network on a continuous stream of data. Recently proposed methods mostly mitigate this challenge by replaying data from previous domains, re-estimating statistics or even in an unsupervised manner by transferring training images in the style of the novel domain [32,48,52]. The focus of our work is to study how the internal representations of se-
semantic segmentation models are affected during domain-incremental learning and how efficient feature reuse can mitigate forgetting without explicit replay of the previous domain. Our main contributions are:

1. We analyze the activation drift that a model’s layers are subjected to when adapting from good to adverse weather conditions by stitching them with the previous task’s network. We reveal that the major cause of forgetting is a shift of low-level representations in the first convolution layer that adversely affects the population statistics of the following BatchNorm Layer.

2. Using different augmentation strategies to match the target domains in color statistics or in the frequency domain, we reveal that learning color-invariant features stabilizes the representations in early layers, as they don’t change when the model is adapted to a new domain.

3. With a combination of pre-training, augmentations and exchanged normalization layers, we achieve an overall reduction of forgetting of $\sim 20\%$ mIoU compared to fine-tuning without using any form of replay and prove the effectiveness of pre-training and augmentations which are often overlooked in continual learning.

2. Related Work

2.1. Continual Work

Continual learning research is primarily concerned with developing methods to overcome catastrophic forgetting. It has been extensively studied in incremental classification tasks, where the approaches can be broadly divided into regularization-based methods [2,14,25,29,58], replay-based methods [19,43,44,47] and parameter-isolation methods [31,50,55]. For semantic segmentation, significant progress has been made on addressing class-incremental learning, mostly utilizing knowledge distillation-based approaches [7,8,15,23,26,34,35,56]. Research on domain-incremental learning for semantic segmentation is relatively sparse. Garg et al. [17] propose a dynamic architecture that learns dedicated parameters to capture domain-specific features for each domain. Mirza et al. [36] circumvent the issues of biased BatchNorm statistics by re-estimating and saving them for every domain, so that during inference domain-specific statistics can be used. Our experiments in this paper show that the effectiveness of these approaches mostly originates in matching the low-level statistics of the initial block of the network. Recently, the focus shifted towards Continual Unsupervised Domain Adaptation (CDA).

2.2. Continual Unsupervised Domain Adaptation

CDA has the goal of adapting a model that is trained on a supervised source dataset to a sequence of different domains, for which no labels are provided. However, in order to compensate for the missing labels of the target domains, the model has access to the initial source dataset throughout the entire training sequence [32]. Methods in this category work mostly by storing information about the style of the specific domains, so that during training the source images can be transferred into the styles of the different target domains. This can be achieved by storing low-frequency components of the domains [48] or by capturing the style using generative models [32,52]. Other recent work proposes to use a target-specific memory for every domain to mitigate forgetting [24]. Although this setting is different from the domain-incremental setting in our work, the insights that we gain on the causes and mitigation strategies, especially the importance of the invariance of low-level features, can be transferred to CDA as well.

2.3. Analysing Catastrophic Forgetting

In previous studies, representational analysis techniques such as centered kernel alignment [27] and linear probing were used to analyze the effects of catastrophic forgetting in deep learning for image classification [13,42]. Their results concluded that deeper layers are disproportionately responsible for forgetting in class-incremental learning. Other work investigates how multi-task- and incremental learning solutions are connected in their loss landscape [38], or how task sequence [40] or task similarity [42] affect catastrophic forgetting. Overall, much of the recent research in analyzing catastrophic forgetting has been primarily focused on classification tasks or class-incremental learning [22,42]. While much progress has been made on mitigating forgetting in domain-incremental semantic segmentation by utilizing style transfer, replay or matching BN statistics, it remains unclear how these specific changes affect the internal representations of the model.

3. Preliminaries

3.1. Problem Formulation

The task of semantic segmentation is to assign a class, out of a set of pre-defined classes $C$, to each pixel in a given image. A training task $T = \{ (x_n, y_n) \}_{n=1}^N$ consists of a set of $N$ images $x \in \mathcal{X}$ with $\mathcal{X} = \mathbb{R}^{H \times W \times 3}$ and corresponding labels $y \in \mathcal{Y}$ with $\mathcal{Y} = \mathbb{C}^{H \times W}$. Given the task $T$ the goal is to learn a mapping $f : \mathcal{X} \mapsto \mathbb{R}^{H \times W \times |C|}$ from the image space $\mathcal{X}$ to a score vector $\hat{y}$. The final segmentation mask is then computed as $\hat{y}_i = \arg \max_{c \in C} \hat{y}_{i,c}$. In the incremental learning setting, the model $f$ is trained on a sequence of tasks $T_i$ that can introduce new classes or visually distinct instances from the same classes. This study focuses on domain-incremental settings, where the input distribution of the data is changing between tasks while the set of classes is fixed. Specifically, we focus on domain-increments in different adverse weather conditions.
3.2. Normalization Layers

Normalization layers are essential in training Convolutional Neural Networks (CNNs), as they address the internal covariate shift of the network by normalizing the inputs to layers, so that the input distributions to each layer are stable during training [20]. Batch Normalization (BN) [20] is the most common CNN normalization layer, which normalizes layer inputs using moments across the mini-batch dimension. However, to achieve deterministic behavior during inference, the mini-batch variance and mean are replaced by the global population mean and variance, which are obtained during training using an exponential running average. This works for i.i.d.\(^1\) data, but in non-i.i.d. incremental learning, the BN estimates of the population mean and variance are heavily biased towards the most recent task, resulting in a significant loss of performance on old tasks [30]. Continual Normalization (CN) alleviates this discrepancy by combining Group- [51] and Batch Normalization [41].

3.3. Measuring Activation Drift

Accuracy-based evaluation only allows for restrictive insight into the causes of forgetting of a model, specifically if representation shifts happen in the early layers of the networks. As we expect catastrophic forgetting to be the result of a change in weights and activations of the model that are no longer tuned to the most recent task, we aim to measure activation drift in incremental learning. Specifically, we want to measure the activation shift between a model \(f_0\) and \(f_1\). Where \(f_0\) denotes the model trained on \(T_0\) and \(f_1\) is the model initialized with the parameters of \(f_0\) and incrementally trained on \(T_1\). For this, we utilize the layer matching framework introduced by Csiszár et al. [12], without an additional stitching layer [22]. Fig. 2 shows the layer stitching setup. In this setup, we measure the impact of activation drift until layer \(n\) by propagating the activations of layer \(n\) directly to the layer \(n+1\) in \(f_0\). The resulting stitching model is then evaluated on the test set of the initial task \(T_0\) and compared to the initial accuracy of the model \(f_0\). This gives us a proxy measure of how useful the features at a specific layer of the adapted model \(f_1\) are for the initial model \(f_0\).

4. Experiments

Datasets: We conduct our experiments on adapting to adverse weather conditions using an domain-incremental setup that involves adapting from the Cityscapes (CS) [11] dataset to ACDC [46], which is commonly used as a benchmark for unsupervised domain adaptation. The CS dataset is an automotive semantic segmentation dataset, collected during daytime and dry weather conditions in different German, Swiss and French cities. It contains 2975 training and 500 validation images. The ACDC dataset is collected during different adverse weather conditions and divided into four different subsets: Night, Rain, Fog and Snow. ACDC and CS share the same 19 classes, so that the changes between the tasks is only based on the domain differences. To study how features are reused and or adapted in each adverse weather condition, we investigate four different scenarios all starting with the same CS model: \(CS \rightarrow Night, CS \rightarrow Rain, CS \rightarrow Snow\) and \(CS \rightarrow Fog\).

Models: We use the widely adopted DeepLabV3+ [9] in our experiments with a ResNet50 backbone, as DeepLabV3+ is a commonly used architecture in domain adaptation. Furthermore, we confirm our findings with different architecture in Appendix F, because the architectural choices can have a significant impact in continual learning [21,37]. Finally, in Sec. 5.2 we show that the recently introduced SegFormer-B2 [53] is more robust towards low-level feature change than its CNN counterparts. In the majority of the experiments, we use randomly initialized models for training on the first task, as pre-training is known to increase robustness to catastrophic forgetting [16,33] by enabling low-level feature reuse, as we will see in Sec. 4.3.

Optimization Strategy We train the networks using SGD as an optimizer with momentum of 0.9 and weight decay of \(3 \times 10^{-3}\). We use a polynomial learning rate schedule with power 0.9, and start CS training with a 0.07 learning rate and use a batch size 8. We use the same learning rate policy for ACDC subset training, starting with \(5 \times 10^{-3}\). CS and ACDC subsets are trained for 200 and 150 epochs, respectively. We crop the images to \(512 \times 1024\) and utilize random horizontal flipping and scaling during training. Unless stated otherwise, we do not use any other augmentations. During testing we do not use any scaling or cropping.

Evaluation Metrics: We evaluate the performance of each model on the validation set of each dataset using the mean intersection-over-union (mIoU). We denote the mIoU of the model trained on all tasks up to \(p\) and evaluated on

\[^1\text{independent and identically distributed}\]
task q as mIoU$_{p,q}$. So that the zero-shot performance of a model trained on task p = 0 and evaluated on q = 1 is denoted as mIoU$_{0,1}$. Furthermore, we report average learning accuracy and forgetting, that measure the learning capability and the severity of forgetting [37].

\[
\text{average learning acc.} = \frac{\text{mIoU}_{0,0} + \text{mIoU}_{1,1}}{2}
\]

\[
\text{forgetting} = \text{mIoU}_{0,0} - \text{mIoU}_{1,0}
\]

### 4.1. Activation Drift after Incremental Adaptation

First, we study the overall activation drift that arises in the CNN models when naively adapting to the different adverse weather conditions. Therefore, we first train DeepLabV3+ on CS and subsequently fine-tune it individually on each of the ACDC subsets. The results are displayed in Tab. 2. As one would expect, the zero-shot performance on the adverse weather conditions is good for visually similar conditions such as Fog and Rain and significantly worse for Snow and Night, with Night being the worst with only 10.4% mIoU$_{0,1}$. This can be explained by the apparent differences between the domains, so that the day-to-night shift and snow-covered landscape represents a bigger shift than the wet environment or the foggy conditions [46]. However, after fine-tuning on the adverse subsets, we notice that the better zero-shot performance on Rain and Fog does not indicate less forgetting compared to Snow and Night. Most strikingly, forgetting is the lowest after adapting to Night. To determine which layers are most affected by the internal activation drift, we measure the activation drift for each layer between the model before and after learning the second task with layer stitching. We use the setup explained in Sec. 3.3. The mIoU relative to the initial performance on the first task is shown in Fig. 1. We observe that activation drift is mainly affecting the network’s early layers, which is in contrast to class-incremental learning settings, where early layers remain stable [13,22,42]. Specifically, the low-level features of the models tuned to Fog and Snow cannot be reused by the initial model, indicating that the shallow layers of the network have changed significantly. However, after the initial drop of relative mIoU in Tab. 2 we see a substantial increase in mIoU after layer1.0, which indicates later features are in fact reused by the Cityscapes model. We hypothesize that at that point, features are more abstract and therefore more useful for the model trained on CS. After layer2.0 we observe a steady decrease in relative mIoU until the decoder layers. We note that once the representations are shifted, subsequent layers are unlikely to regain similarity as their representations are based on the output of the previous layer. The changing image distribution is most likely to blame for the initial feature disparity. Thus, we analyze domain image distribution at the pixel level in the next section.

![Amplitude Spectrum](image)

**Figure 3.** Amplitude Spectra of Cityscapes, augmented CS and the ACDC subsets. In the frequency domain Cityscapes is much more similar to Night than to any other of the ACDC subsets, specifically in the high-frequent components of the images. Blur is efficiently removing of high frequency components.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Hue</th>
<th>Saturation</th>
<th>Value</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rain</td>
<td>64</td>
<td>122</td>
<td>60</td>
<td>64, 64, 46</td>
</tr>
<tr>
<td>Fog</td>
<td>93</td>
<td>20</td>
<td>131</td>
<td>60, 23, 64</td>
</tr>
<tr>
<td>Cityscapes</td>
<td>59</td>
<td>49</td>
<td>83</td>
<td>18, 22, 49</td>
</tr>
<tr>
<td>Snow</td>
<td>104</td>
<td>19</td>
<td>132</td>
<td>55, 21, 62</td>
</tr>
<tr>
<td>Night</td>
<td>64</td>
<td>122</td>
<td>60</td>
<td>64, 64, 46</td>
</tr>
</tbody>
</table>

Table 1. The Mean and standard deviations for the HSV-channels of each subset. There is a severe color shift between the domains in overall brightness of the images.

### 4.2. Analysis of image statistics

To interpret the low-level feature change within the network, we compare the image statistics of each domain to explain the significant representation changes in the early layers. Therefore, we compare the domains by their corresponding pixel mean and standard deviation for each HSV-channel in Tab. 1. We observe that there is a substantial difference between the domains, specifically Rain, Snow and Fog being notably brighter than Cityscapes. Furthermore, as the capability of CNNs to generalize can be adversely affected by exploiting mid- to high-frequency components of images [1,49], we also analyze the amplitude spectra in the frequency domain of the different training tasks in Fig. 3. We can see that the domains are similar in low-to mid-frequency ranges, but Snow and Rain contain much more high-frequency components. This could lead to over-fitting to high-frequency features when the model is trained on these domains and magnify forgetting.

### 4.3. Adjusting Low-Level Features

Previous work suggests that low-level feature reuse is important for successful transfer learning [39] and that pre-training can mitigate forgetting for class-incremental classification tasks [16,33]. In a set of experiments, we show that low-level feature reuse is not only important for success-

![Table 1](image)
ful transfer of knowledge to downstream tasks, but is also vital to reduce catastrophic forgetting. Therefore, we investigate different pre-training and augmentation protocols to initialize the model \( f_0 \) on CS. We then fine-tune the model \( f_0 \) on the different ACDC subsets without any augmentations or any continual learning algorithm. Thus, we can examine how different methods improve the model’s feature reuse and reduce forgetting. To study the impact on pre-trained backbones, we use ResNet50 weights trained on ImageNet1k either fully-supervised or using the self-supervised methods DINO [6], MoCo v3 [10], SwAV [5] and BarlowTwins [57]. For the augmentation experiments, we use the following strategies:

- Using AutoAlbument (AutoAlbum) [4], to learn an image augmentation policy from the CS dataset using Faster AutoAugment [18].
- Learning color-invariant features by intensive Color Jittering and randomly rearranging input image channels. We denote this combination as Distortion (Distort).
- Learning features tuned for mid- to low-frequencies by Gaussian blurring or adding Gaussian noise to remove high-frequency information. Changes to the spectrum are displayed in Fig. 3.

We also add an offline pre-trained model that is trained jointly on the CS and ACDC subsets and then fine-tuned on the target task. We use offline pre-training to infer an upper bound on a model’s feature reuse, as the model should have learned features that are the joint optimum on both tasks. The results are displayed in Tab. 3. The overall trend we see is that both pre-training and augmentations during training on CS lead to better transfer to the subsequent tasks and reduced forgetting on CS. However, while pre-training improves the transfer to the new tasks, it only moderately improves zero-shot capabilities when compared to the model without pre-training. Color-based augmentation and AutoAlbum improve zero-shot performance but perform worse on the ACDC tasks than the pre-trained models, indicating that better zero-shot performance does not always lead to better transfer performance. Still, these augmentations are the most effective at mitigating forgetting for all tasks. Augmentations based on removing high-frequency components reduce forgetting less significantly or, in the case of Fog, lead to a further decrease in mIoU on the previous task. We note that for the domains Rain and Snow that contain more high-frequency components in their images than CS, adding noise and blurring is more effective than it is for Fog and Night. So far, the results indicate two things: 1) pre-trained features are less susceptible to forgetting and lead to a better transfer to future tasks, 2) augmentations significantly improve generalization and produce more general features in the early layers.

We also investigate these findings using layer stitching and display the plots in Fig. 4. The pre-trained models (bottom row) have a lower initial drop in similarity than randomly initialized models and remain higher throughout later layers. Noticeable is that even the offline pre-trained model is affected by a severe drop in similarity for Snow, Fog and also a moderate drop in similarity for Rain and Fog. We will later confirm that this is largely due to the biased population mean and standard deviation of the BN layers. Most surprising is that the models that used color augmentation during training are not affected by this initial drop, even though the optimization process on task 2 is the same for the offline and pre-trained models, so they should be affected by the same change in population statistics. The fact that this does not occur indicates that when training with augmentations, the first convolutional layers extract features that are more domain-invariant than the features of the pre-trained models, resulting in BN layer population statistics that are less biased to the previous task [41]. We validate this claim in the next section.

### 4.4. Impact of Batch Normalization on Forgetting

The results in Sec. 4.3 suggest that changing BN population statistics are a major cause of early layer representation changes. To verify this, we re-estimate the BN Layer population statistics on the combined dataset of CS and the specific ACDC subset without changing any parameters.\(^2\) We then evaluate the model on the CS dataset and report the re-estimated mIoU and increased performance as \(\Delta\)mIoU in Tab. 5. Most methods benefit significantly from re-estimation of population statistics, with the Fine-Tuning (FT) model benefiting the most. Furthermore, we observe that pre-trained models improve only moderately compared to FT, meaning that they are less influenced by biased population statistics. Finally, we discover that models trained with augmentation only slightly improve after BN re-estimation, and even decrease in the CS \(\rightarrow\) Fog setting. We explain this effect by the invariance to low-level properties of the images such as hue, saturation and brightness that the first CNN layer of the models trained with augmentations has to learn in order to cope with the augmentation.

\(^2\)This is achieved by doing a forward pass over the dataset while the BN Layers update their population statistics
<table>
<thead>
<tr>
<th>Method</th>
<th>Cityscapes Test mIoU</th>
<th>Night Test mIoU</th>
<th>Rain Test mIoU</th>
<th>Fog Test mIoU</th>
<th>Snow Test mIoU</th>
</tr>
</thead>
<tbody>
<tr>
<td>FT</td>
<td>72.0</td>
<td>43.6</td>
<td>26.1</td>
<td>33.2</td>
<td>28.0</td>
</tr>
<tr>
<td>AutoAlb.</td>
<td>72.2</td>
<td>47.3</td>
<td>15.1</td>
<td>10.7</td>
<td>14.7</td>
</tr>
<tr>
<td>Distort</td>
<td>71.7</td>
<td>46.5</td>
<td>16.2</td>
<td>6.5</td>
<td>15.2</td>
</tr>
<tr>
<td>Noise</td>
<td>69.1</td>
<td>46.3</td>
<td>23.0</td>
<td>6.9</td>
<td>28.3</td>
</tr>
<tr>
<td>Noise</td>
<td>69.8</td>
<td>46.7</td>
<td>21.3</td>
<td>6.9</td>
<td>30.2</td>
</tr>
<tr>
<td>ImageNet</td>
<td>73.9</td>
<td>47.5</td>
<td>22.5</td>
<td>21.8</td>
<td>23.8</td>
</tr>
<tr>
<td>MOCO</td>
<td>75.2</td>
<td>48.3</td>
<td>17.3</td>
<td>41.9</td>
<td>22.3</td>
</tr>
<tr>
<td>DINO</td>
<td>75.0</td>
<td>49.7</td>
<td>18.7</td>
<td>6.4</td>
<td>19.6</td>
</tr>
<tr>
<td>BarlowT.</td>
<td>73.9</td>
<td>47.3</td>
<td>17.1</td>
<td>65.7</td>
<td>14.4</td>
</tr>
<tr>
<td>SwAV</td>
<td>76.4</td>
<td>48.1</td>
<td>17.8</td>
<td>62.4</td>
<td>24.5</td>
</tr>
<tr>
<td>Offline</td>
<td>-</td>
<td>46.1</td>
<td>3.2</td>
<td>59.0</td>
<td>3.2</td>
</tr>
</tbody>
</table>

Table 3. Results on CS → ACDC in mIoU (%) for each subset of ACDC: Night, Rain, Snow and Fog using different pre-training and augmentation strategies (Augment.). While pre-training significantly improves learning accuracy, it does not improve zero-shot performance, but still gives moderate improvements in reducing forgetting. Augmentations lead to slightly improved performance on the target set, improved zero-shot performance and significantly reduced forgetting for all weather conditions.

![Figure 4. Activation drift between $f_1$ to $f_0$ measured by relative mIoU on the first task of the model stitched together at specific layers (horizontal axis). The layers of the encoder are marked in the gray area, the decoder layers in the white area. The activations up until layer 1.0 undergo drastic changes, specifically for Snow and Fog. After layer 1.0 the activations can again be reused by $f_0$ leading to an mIoU increase. However, throughout the remaining encoder layers of the network the activations of $f_1$ further deviate from $f_0$.](image)

3 In Appendix C, we investigate which BN layers are affected.
We run the same experiments making incremental changes to the training process in CS by sequentially adding pre-training with DINO, then AutoAlbum and replacing BN with CN layers. Results in Tab. 6 demonstrate that these changes complement each other as they dramatically reduce forgetting on CS. This is apparent when comparing the layer stitching plots in Fig. 6, where we see that pre-training with DINO alone increases feature reuse only after layer1.0 compared to fine-tuning. However, when combined with augmentations and CN, the representation drift before layer1.0 is significantly reduced as well. This indicates that pre-training and training with augmentations enable feature reuse at different layers of the network depending on the task at hand. Therefore, when combining pre-training, CN and augmentation, we increase the feature reuse in all domains, as we see in Fig. 1, reducing forgetting without any continual learning algorithm.

5. Ablation Studies

5.1. Comparison to Continual Learning Algorithms

So far, no explicit continual learning strategies like regularization or replay were used. Therefore, we show in Tab. 7 that achieving low-level feature reuse outperforms regularization methods even when using the same initialization. However, our training regime is still outperformed by naive replay. While EWC significantly improves when combined with CN and AutoAlbum, we see that Replay does not benefit from these adjustments. For Replay only pre-training leads to a significant increase in learning accuracy and a minor reduction of forgetting. A likely explanation is that the model is able to learn features that are invariant to domain differences due to the batch construction during replay, in which half of the mini-batch consists of replay samples from previous tasks. This could explain why replay is reported to be sample efficient in domain incremental learning [23].

5.2. Architectures

We confirm our findings by repeating our experiments on SegFormer [53], DeepLabV3+, and ERFNet [45], all of which were pre-trained on ImageNet. We choose DeepLabV3+ with a ResNet50 backbone and SegFormer-
6. Conclusion

Our study has shown that a major cause of catastrophic forgetting in domain-incremental learning is the shift of low-level representations, particularly in the first convolution layer. This shift affects the population statistics of subsequent BN layers and results in forgetting when adapting to new domains. To address this problem, we investigated the use of various pre-training schemes and pixel-level augmentations to facilitate features in early layers that can be reused in upcoming tasks. Our experiments showed that these methods were effective in reducing representation shift, with pre-training stabilizing the first layers, and augmentations primarily stabilizing the representations after the first BN layer. We hypothesize that training with augmentation strategies like Distortion or AutoAlbum encourages the model to learn features that are invariant to low-level image statistics such as hue, saturation and brightness that vary between the domains. So that during optimization on the new domain those features are not affected, leading to a significant reduction in forgetting. Interestingly, we found that pre-trained models struggle to learn such features in the early layers, but they still reduce forgetting notably compared to randomly initialized models. Leading us to believe that pre-training on ImageNet leads to more generalized features throughout the network. In our experiments, self-supervised pre-training outperformed supervised Imagenet pre-training on all domains, which suggests that SSL pre-training might not only be a vital tool for classification [16], but also for continual semantic segmentation. The effectiveness of these approaches varies across the domains but is consistent for different CNN architectures that use BN layers\(^4\). Our findings are related to the research on spurious features in continual learning [28], as our training scheme is reducing the emergence of spurious features for the first task. Overall, we hope that our results highlight that an important component of continual learning can be found in methods that extract generalized features from the initial task, instead of only mitigating the effects of catastrophic forgetting during training on new data.

However, we note that even with improved low-level feature reuse, the model is still susceptible to catastrophic forgetting in later layers due to more abstract domain changes. As a result, we consider low-level feature reuse in incremental learning to be an important component of continual learning, but not the sole solution.

ACKNOWLEDGMENT

The research leading to these results is funded by the German Federal Ministry for Economic Affairs and Climate Action within the project “KI Delta Learning” (Förderkennzeichen 19A19013T).

\(^4\)Experiments on architectures are displayed in Appendix F
References


[23] Tobias Kalb, Masoud Roschani, Miriam Ruf, and Jürgen Beyerer. Continual learning for class- and domain-


