

# **Principles of Catastrophic Forgetting for Continual Semantic Segmentation in Automated Driving**

Zur Erlangung des akademischen Grades eines  
Doktors der Ingenieurwissenschaften

von der KIT-Fakultät für Informatik  
des Karlsruher Instituts für Technologie (KIT)

**genehmigte  
Dissertation**

von

M.Sc.

**Tobias Michael Kalb**

aus Hünfeld

Tag der mündlichen Prüfung:  
Erster Gutachter:  
Zweiter Gutachter:

19.02.2024  
Prof. Dr.-Ing. habil. Jürgen Beyerer  
Prof. Dr. Hanno Gottschalk



# Abstract

Current deep learning algorithms are capable of learning from large amounts of data and extracting complex patterns. They have been successful in various applications such as computer vision, natural language processing and speech recognition. A central challenge in integrating deep learning systems into complex real-world applications that involve a dynamic and evolving environment, like automated driving, is the phenomenon of catastrophic forgetting. Catastrophic forgetting refers to the tendency of deep learning systems to completely forget previously learned information when trained on new data. This poses a significant challenge in the development of reliable and robust deep learning models for real-world applications. While research in continual learning aims to overcome catastrophic forgetting and enable deep learning models to retain and adapt to new information without losing previously acquired knowledge, the underlying principles and effects of catastrophic forgetting remain mostly obscure.

Therefore, instead of pursuing incremental algorithmic improvements to continual learning methods, this thesis aims to reveal the causes and consequences of catastrophic forgetting that arise during continual learning and that remain valid independently of the progress of the state-of-the-art methods. Through an exploration of continual learning scenarios for real-world semantic segmentation in automated driving, distinctive characteristics of forgetting that arise in class- and domain-incremental semantic segmentation are uncovered. To investigate these characteristics further, a set of analysis tools that allow quantitatively measuring the effects of catastrophic forgetting are compared. Utilizing these methods, the principles of forgetting are studied independently for class- and domain-incremental learning. It is found that in class-incremental learning, the semantic shift of the background class

is a major cause of forgetting, which primarily affects the layers close to the output layer. In contrast, in domain-incremental learning, forgetting is often caused by changes to low-level features that primarily affect the input layers of the network. Furthermore, it is found that methods such as strong image augmentations or comprehensive pre-training that lead to more generalized features, enable the model to reuse learned features in future tasks, which drastically reduces catastrophic forgetting. Finally, the effects of several architectural choices are studied, revealing that the increased robustness towards catastrophic forgetting of vision transformers is mainly caused by the better generalization capabilities of these models. However, this increased robustness can be replicated in convolutional neural networks by modifying architectural elements such as increasing kernel sizes, exchanging normalization layers and input transformations.

Overall, this thesis provides insight into the intricate mechanisms of catastrophic forgetting in continual learning scenarios, shedding light on the causes and effects that impact the performance of deep learning models in complex real-world applications. These findings hold promise for the development of more resilient and adaptable deep learning systems, crucial for ensuring the reliability and effectiveness of applications like automated driving.

# Kurzfassung

Aktuelle Deep-Learning-Algorithmen sind in der Lage, aus großen Datenmengen zu lernen und komplexe Muster zu extrahieren. Sie haben sich in verschiedenen Anwendungen wie Computer Vision, Textverarbeitung und Spracherkennung bewährt. Eine zentrale Herausforderung für die Integration von Deep-Learning-Systemen in dynamischen und sich verändernden realen Anwendungen, wie z. B. automatisiertes Fahren, ist das sogenannte "katastrophale Vergessen". Katastrophales Vergessen bezeichnet die Tendenz von Deep-Learning-Systemen, zuvor gelernte Informationen vollständig zu vergessen, wenn sie mit neuen Daten trainiert werden. Dies stellt eine große Herausforderung bei der Entwicklung zuverlässiger und robuster Deep-Learning-Modelle für reale Anwendungen dar. Obwohl die Forschung im Bereich des kontinuierlichen Lernens darauf abzielt, das katastrophale Vergessen zu überwinden und Deep-Learning-Modelle in die Lage zu versetzen, neue Informationen zu lernen und sich an diese anzupassen, ohne das zuvor erworbene Wissen zu verlieren, sind die zugrundeliegenden Prinzipien und Auswirkungen des Vergessens noch relativ unklar.

Deshalb zielt diese Arbeit nicht darauf ab, die Methoden des kontinuierlichen Lernens inkrementell zu verbessern, sondern die Ursachen und Folgen des katastrophalen Vergessens aufzudecken, die während des kontinuierlichen Lernens auftreten und unabhängig von den Fortschritten der modernsten Methoden gültig bleiben. Durch die Analyse von klassen- und domäneninkrementellen Lernszenarien für die reale semantische Segmentierung beim automatisierten Fahren werden charakteristische Merkmale des Vergessens, die bei klassen- und domäneninkrementeller semantischer Segmentierung auftreten, aufgedeckt. Um diese Charakteristika weiter zu untersuchen, werden eine Reihe von Analysewerkzeugen verglichen, die es erlauben, die Auswirkungen des

katastrophalen Vergessens quantitativ zu messen. Mit Hilfe dieser Methoden werden die Prinzipien des Vergessens separat für klassen- und domäneninkrementelles Lernen untersucht. Es zeigt sich, dass beim klasseninkrementellen Lernen die semantische Änderung der Hintergrundklasse eine Hauptursache für das Vergessen ist, was vor allem die Schichten des neuronalen Netzes nahe der Ausgabeschicht betrifft. Im Gegensatz dazu wird das Vergessen beim domäneninkrementellen Lernen häufig durch Änderungen von Low-Level-Merkmalen verursacht, die in erster Linie die Eingabeschichten des Netzes betreffen. Darüber hinaus zeigt sich, dass Methoden wie starke Bildaugmentierungen oder umfassendes Pre-Training, die generalisiertere Merkmale ermöglichen, das Modell in die Lage versetzen, gelernte Merkmale in zukünftigen Aufgaben wiederzuverwenden, wodurch katastrophales Vergessen drastisch reduziert wird. Schließlich werden die Auswirkungen verschiedener Architekturen untersucht, wobei sich herausstellt, dass die erhöhte Robustheit gegenüber katastrophalem Vergessen von Vision-Transformern hauptsächlich auf die besseren Generalisierungsfähigkeiten dieser Modelle zurückzuführen ist. Diese gesteigerte Robustheit kann jedoch in faltbaren neuronalen Netzen durch Modifikation von Architekturelementen wie Erhöhung der Kernelgröße, Austausch von Normalisierungsschichten und Eingangstransformationen repliziert werden.

Die vorliegende Arbeit untersucht die vielschichtigen Mechanismen des katastrophalen Vergessens in kontinuierlichen Lernszenarien und liefert Erkenntnisse über die Ursachen und Auswirkungen, die die Leistung von Deep-Learning-Modellen in komplexen realen Anwendungen beeinflussen. Diese Erkenntnisse sind vielversprechend für die Entwicklung robusterer und anpassungsfähigerer Deep-Learning-Systeme, die für die Gewährleistung der Zuverlässigkeit und Effektivität von Anwendungen wie dem autonomen Fahren entscheidend sind.

# Acknowledgements

This thesis was written during my work as a doctoral student in the department of Big Data and AI at Porsche Engineering Services GmbH and under the supervision of Prof. Dr.-Ing. Jürgen Beyerer, head of the Vision and Fusion Laboratory at the Karlsruhe Institute and Technology, as well as head of the Fraunhofer Institute of Optronics, System Technologies and Image Exploitation IOSB.

First of all, I would like to express my sincere gratitude to Prof. Dr.-Ing. Jürgen Beyerer for his continuous support and guidance. His expertise and mentorship have been essential throughout my research journey. In addition, I would like to thank Prof. Dr. Hanno Gottschalk and T.T.-Prof. Dr. Pascal Friederich for their cooperativeness and their willingness to act as a second reviewer and examiner.

My sincere and deepest gratitude goes to Jens Ziehn, Masoud Roschani, and Dr.-Ing Miriam Ruf, who served as not only mentors but also invaluable technical supervisors throughout this dissertation. Their guidance, expertise, and constant support have been instrumental in shaping the research presented in this work. Thank you for being encouraging and inspiring, for being open to my ideas and concerns, and especially for your commitment in the days before submission deadlines.

Furthermore, I want to thank my supervisor at Porsche Engineering, Dr. rer. nat. Joachim Schaper. You were an great mentor in many aspects of my studies and taught me valuable skills for navigating the corporate world. Your guidance and insights have not only enriched my academic experience but also prepared me for the challenges and opportunities that lie ahead in my career.

My sincere thanks also extend to my colleagues at Porsche Engineering, especially Jingxing Zhou and Daniel Schumacher. Your valuable contributions to our discussions and constant assistance have been fundamental to the success of my research. More than anything, I thank you for the wonderful environment you helped create, which not only makes our working hours productive but also enjoyable.

A special thanks goes to Jens Ziehn, Katrin Kalb, Jingxing Zhou, and Celina Würfl, who did parts of the proofreading of this thesis and provided valuable comments to increase its overall quality.

I would also like to express my heartfelt appreciation to my friends and family in Kirchhasel, Stuttgart and Esslingen. Your support and shared moments of joy have been invaluable during this academic endeavor.

I'm immensely grateful to Celina, whose unwavering encouragement made even the most stressful situations bearable. Your support has been a constant source of strength and comfort throughout this journey.

The greatest thanks belong to my parents for their years of support and encouragement in my life. You are the anchor in my life where I find strength for new challenges, and for that, I thank you.

# Contents

<b>Abstract</b> . . . . .	<b>i</b>
<b>Kurzfassung</b> . . . . .	<b>iii</b>
<b>Acknowledgements</b> . . . . .	<b>v</b>
<b>Notation</b> . . . . .	<b>xiii</b>
<b>1 Introduction</b> . . . . .	<b>1</b>
1.1 Continual Learning in Highly Automated Driving . . . . .	2
1.2 Why Continual Learning in Automated Driving? . . . . .	5
1.3 Contributions . . . . .	7
1.4 Thesis Outline . . . . .	10
<b>2 Background</b> . . . . .	<b>11</b>
2.1 Artificial Neural Networks . . . . .	11
2.2 Deep Neural Networks for Computer Vision . . . . .	13
2.2.1 Convolutional Neural Networks . . . . .	13
2.2.2 Vision Transformers . . . . .	18
2.3 Continual Learning . . . . .	20
2.3.1 Catastrophic Forgetting . . . . .	22
2.3.2 Desiderata . . . . .	24
2.4 Semantic Segmentation . . . . .	26
2.5 Evaluation Metrics . . . . .	27
2.5.1 Semantic Segmentation . . . . .	28
2.5.2 Continual Learning Metrics . . . . .	30

<b>3</b>	<b>Related Work</b>	<b>33</b>
3.1	Continual Learning Approaches	33
3.1.1	Replay-Based Methods	34
3.1.2	Regularization-Based Methods	35
3.1.3	Parameter Isolation Methods	37
3.2	Continual Semantic Segmentation	38
3.2.1	Class-Incremental Semantic Segmentation (CiSS)	38
3.2.2	Domain-Incremental Semantic Segmentation (DiSS)	40
3.3	Effects of Catastrophic Forgetting	41
<b>4</b>	<b>Continual Learning for Class- and Domain-Incremental Semantic Segmentation</b>	<b>43</b>
4.1	Incremental Semantic Segmentation	44
4.1.1	Domain-Incremental Segmentation	45
4.1.2	Class-Incremental Segmentation	46
4.2	Experimental Setup	47
4.2.1	Baseline Overview	48
4.2.2	Evaluation	50
4.3	Results and Discussion	51
4.3.1	Results on Class-Incremental Learning	51
4.3.2	Results on Domain-Incremental Learning	54
4.3.3	Comparison to Results on Image Recognition	55
4.4	Improving Data Selection for Replay	56
4.4.1	Sample Selection for Domain-Incremental Learning	57
4.4.2	Sample Selection for Class-Incremental Learning	58
4.4.3	Different Buffer Sizes	59
4.5	Conclusion and Research Questions	61
<b>5</b>	<b>Methods to Measure the Causes and Effects of Catastrophic Forgetting</b>	<b>63</b>
5.1	Overview	63

5.2	Experimental Setup . . . . .	65
5.3	Activation Drift . . . . .	66
5.3.1	Layer Matching with Dr. Frankenstein . . . . .	66
5.3.2	Centered Kernel Alignment (CKA) . . . . .	72
5.4	Re-Training and Re-Estimation . . . . .	75
5.4.1	Batch Normalization Re-Estimation . . . . .	75
5.4.2	Partial Re-Training Accuracy (PRA) . . . . .	77
5.4.3	Decoder Re-Train Accuracy and Linear Probing . . . . .	78
5.5	Weight Drift . . . . .	79
5.6	Conclusion . . . . .	82
<b>6</b>	<b>Principles of Catastrophic Forgetting in Continual Semantic Segmentation . . . . .</b>	<b>85</b>
6.1	Principles of Catastrophic Forgetting in Class-Incremental Learning . . . . .	86
6.1.1	Experimental Setup . . . . .	87
6.1.2	Semantic Background Shift and Class Confusion . . . . .	89
6.1.3	Activation Drift in Class-Incremental Learning . . . . .	93
6.1.4	The Impact of Inter-task Confusion on the Encoder . . . . .	95
6.1.5	Reducing Background Bias and Task Recency Bias . . . . .	97
6.1.6	Effects of Different Architectures . . . . .	99
6.1.7	The Role of the Background Class to Overcome Forgetting . . . . .	102
6.1.8	Conclusion . . . . .	103
6.2	Principles of Catastrophic Forgetting in Domain-Incremental Learning . . . . .	103
6.2.1	Related Work . . . . .	104
6.2.2	Experimental Setup . . . . .	105
6.2.3	Activation Drift after Incremental Adaptation . . . . .	107
6.2.4	Analysis of Image Statistics . . . . .	109
6.2.5	Adjusting Low-Level Features . . . . .	111
6.2.6	Impact of Batch Normalization on Forgetting . . . . .	116

6.2.7	Layer Freezing Experiments . . . . .	119
6.2.8	Combining the Findings . . . . .	121
6.2.9	Ablation Studies . . . . .	123
6.2.10	Conclusion . . . . .	128
6.3	Differences in Domain- and Class-Incremental Learning . . . . .	129
<b>7</b>	<b>Effects of Architecture in Continual Learning . . . . .</b>	<b>133</b>
7.1	Related Works . . . . .	134
7.1.1	Differences between CNNs and Vision Transformers . . . . .	134
7.1.2	Architectures in Continual Learning . . . . .	134
7.2	Experiments . . . . .	135
7.2.1	Experimental Setup . . . . .	136
7.2.2	Domain-Incremental Learning . . . . .	139
7.2.3	Class-Incremental Learning . . . . .	143
7.3	Ablation on ConvNeXt . . . . .	147
7.4	Conclusion . . . . .	150
<b>8</b>	<b>Conclusion and Outlook . . . . .</b>	<b>151</b>
8.1	Conclusion . . . . .	151
8.2	Outlook . . . . .	154
	<b>Bibliography . . . . .</b>	<b>157</b>
	<b>Publications . . . . .</b>	<b>185</b>
	<b>List of Figures . . . . .</b>	<b>187</b>
	<b>List of Tables . . . . .</b>	<b>191</b>
	<b>Acronyms . . . . .</b>	<b>195</b>

---

## Appendix

<b>A</b>	<b>Confusion Matrices for the Pascal-15-5 Overlapped Setting</b>	<b>199</b>
<b>B</b>	<b>Reproducibility of Sec. 6.1</b>	<b>201</b>
B.1	Training Protocol	201
B.2	Continual Hyperparameter Selection	201
<b>C</b>	<b>Reproducibility of Sec. 6.2</b>	<b>203</b>
C.1	Training Protocol	203
C.2	Models and Weights	204
C.3	Augmentations	204
<b>D</b>	<b>Reproducibility of Ch. 7</b>	<b>205</b>
D.1	Training Protocol	205
D.2	Models and Weights	205



# Notation

This chapter introduces the notation and symbols which are used in this thesis.

## General Defintions

$\epsilon$	infinitesimally small positive quantity
$\mu$	arithmetic mean
$\sigma^2$	variance
$\ \cdot\ _F$	Frobenius Norm
$\ \cdot\ _2$	Euclidean Norm

## Datasets and Tasks

$T$	training task
$S$	A disjoint subset of a dataset
$k$	index of current task
$x_m$	image of dataset
$y_m$	label of dataset
$p_m$	sample probability of training sample $(x_m, y_m)$
$\mathcal{C}_k$	set of classes of the current task
$C_k$	number of classes of the current task
$\mathcal{S}_k$	set of novel classes that are introduced in task $T_k$

$\mathcal{C}_{k-1} \dot{\cup} \mathcal{S}_k$	the disjoint union of the sets $\mathcal{C}_{k-1}$ and $\mathcal{S}_k$
$I$	Set of pixels in an image

## Artificial Neural Networks

$f_k$	model after optimization on task $T_k$
$N$	number of the layers of model $f$
$\theta_k$	parameters of model $f_k$
$l_{k,n}$	layer $n$ of model $f_k$
$\mathcal{A}_n$	activations space $l_n$
$\mathbf{A}_{k,n}$	activation / feature map of $l_{k,n}$ with dimensions $H \times W \times C$
$\mathcal{L}$	loss function
$H$	height of a feature map
$W$	width of a feature map
$C$	number of channels of a feature map
$B$	batch dimension
$\tau$	temperature parameter of the softmax function

## Numbers and indexing

$i, j, v$	indices
$c$	indexing for classes
$\mathbb{R}$	real numbers

# 1 Introduction

The world around us is highly complex, constantly changing and evolving. To cope with this complexity, humans possess a unique ability to continually learn and adapt. We successfully navigate through diverse environments, recognize objects, understand language, and make sense of the world. In today's technological-driven society, we are challenged to transfer these abilities to computers to automate and solve complex problems without human intervention over long-term continuous operation. However, fully comprehending the intricate workings of these capabilities remains a challenge, making it difficult to directly replicate them in machines

Previous advances in machine learning allow us to partially overcome this restriction by allowing machines to learn from large amounts of data that enable them to infer patterns and to solve complex tasks for which human experts fail to develop problem-specific algorithms. Specifically, the advances in deep learning and neural networks have enabled machines to mimic human learning capabilities in certain domains, such as computer vision and natural language processing. Over the past decades, the capabilities of deep neural networks have increased substantially from only classifying hand written digits in 1989 [LeC89], to image recognition [Den09], to translating and understanding natural language [Dev18] and even surpassing human-level performance in strategic games like chess [12], Go [16] and even particular instances of computer vision such as traffic sign recognition [Cir12] or scene perception [Lon15].

However, contrary to our ability to learn continually from new experiences by refining existing knowledge and accumulating new knowledge, the training process of these deep neural networks currently produces static models that cannot easily be expanded in their function without adversely interfering

with their ability to perform previously-learned tasks. This limitation hinders the adaptability and flexibility of neural networks, as they require extensive retraining for each new task.

Research in the field of *continual learning* aims to overcome this limitation by enabling machines to learn continually from new data while preserving previously acquired knowledge. Continual learning encompasses the ability to incrementally acquire new skills, refine existing ones, and seamlessly integrate new information into the existing knowledge framework.

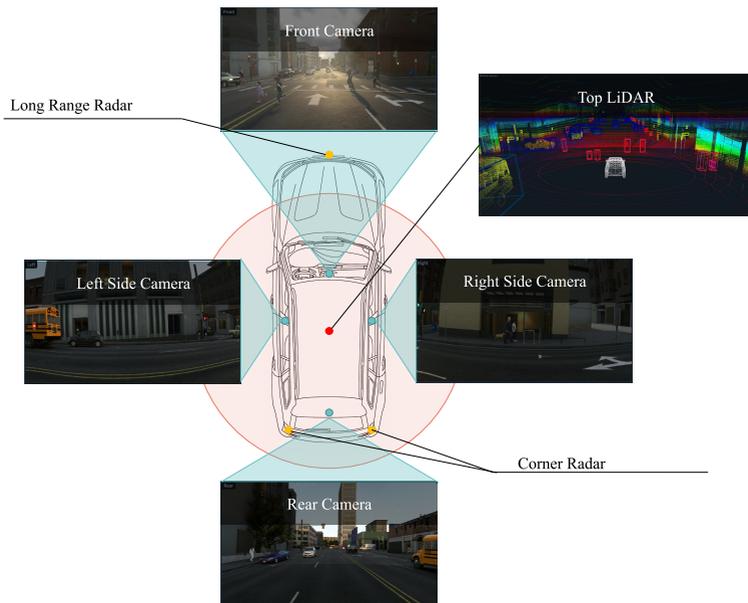
However, progressively acquiring new knowledge without adversely interfering with previously learned knowledge is one fundamental constraint of state-of-the-art deep learning algorithms. This challenge is known as *catastrophic forgetting* [McC89]. Catastrophic forgetting refers to the phenomenon where a neural network's performance on previously learned tasks significantly degrades when it is trained on new data.

The objective of this thesis is to investigate the principles of catastrophic forgetting for deep neural networks in continual learning. Although the focus within this thesis is on scene perception for highly automated driving, many of the observations are likely to be applicable beyond this specific application.

## 1.1 Continual Learning in Highly Automated Driving

Over the past decade, remarkable progress has been made in the field of automated driving, from the first automated vehicles in the DARPA Urban Challenge [Bue09] to now having fully autonomous robotaxis available in selected cities around the globe [Ama21, Vij22, Che23] and the widespread adoption of advanced driver assistance systems such as adaptive cruise control, automatic emergency braking system or lane keeping assist that are fading into the transition to fully autonomous systems. These achievements are promising to revolutionize transportation by enhancing road safety, optimizing traffic flow, and providing greater accessibility and efficiency. Central to the success of

automated driving systems is the ability of their perception systems to accurately perceive and interpret the surrounding environment. These systems are tasked with interpreting and recognizing the vehicle's surroundings by detecting pedestrians, traffic signs, road markings, and other vehicles using measurements obtained by the vehicle's sensor suite, which most commonly consists of cameras, Light Detection and Ranging (LiDAR) and Radio Detection and Ranging (radar) sensors. An exemplary setup of such a system is displayed in Fig. 1.1. The raw data obtained with such a comprehensive sensor suite has then to be processed and interpreted by the perception system in real-time to make safe and reasonable driving decisions. Scene perception, particularly semantic segmentation, plays a critical role in understanding the complex visual information that an autonomous vehicle encounters.



**Figure 1.1:** Illustration of a typical sensor suite for an exemplary automated vehicle.

Recent advances in deep learning have significantly enhanced the capabilities of scene perception systems, enabling automated vehicles to extract high-level semantic information from raw sensor data with unprecedented accuracy and efficiency. Deep neural networks, in particular, have emerged as powerful tools for solving complex computer vision tasks, providing a promising solution for scene perception in automated driving. However, state-of-the-art deep learning approaches rely on full supervision in the form of staggering amounts of manually-labeled image collections, to infer patterns and learn a robust visual recognition model. Even then, when large amounts of training data are available, current learning algorithms require that the training data exhaustively represent what the model will encounter in the real world, since it cannot account for yet unseen requirements. This poses a considerable limitation to using deep learning for perception in automated driving, as it is virtually impossible to collect data for requirements that will arise in the future.

Therefore, it is expected that the perception system will require updates to adapt to the changing driving environment, e.g. for the adaptation of these driving systems to a different market that introduces a significantly different driving environment or to classify new objects that were previously unknown, such as e-scooters or motor-rickshaws. The most efficient way to update such a system would be an incremental update in which the model is trained exclusively on new data while incorporating the new information into the knowledge it has already acquired in previous training sessions. This would not only significantly reduce the time until such an update would be available but also the computational cost for such an update, as it only requires training on the new data and also decreases the dependency on owning and storing all data.

However, this repeated updating process will inevitably lead to catastrophic forgetting. In the context of continual semantic segmentation in automated driving, catastrophic forgetting poses a significant obstacle. Currently, the only way to circumvent this limitation is to completely retrain the models on all available training data. This leads to a significant overhead as every minor update would require a complete retraining of the model, instead of just an incremental update on the new data. An even bigger challenge is that data are not always collectively available due to data privacy restrictions or

data storage capacity or ownership of the data. For this reason, continual learning aims to establish training methods that overcome the challenge of catastrophic forgetting, in which a model forgets the knowledge that it has learned from previous tasks while learning a new task.

In recent years, many algorithms were proposed to overcome catastrophic forgetting for incremental nominal recognition tasks. The performance of these algorithms is mostly measured using only the accuracy of the models on the test sets of the tasks in the sequence. While the models' accuracy on the test set is useful for comparing different learning algorithms on their respective benchmarks, it cannot provide insight into how and where a deep learning model is affected by catastrophic forgetting.

The objective of this thesis is to investigate the principles of catastrophic forgetting for continual semantic segmentation that can arise in an automated driving context. Contrary to prior work, the focus is not on incremental improvements to continual learning algorithms but instead on investigating and understanding the underlying causes and effects of catastrophic forgetting on neural networks that are expected to remain valid independently of the progress of state-of-the-art methods. The insights gained are intended to aid in the development of more robust and adaptable perception models, which will be essential for the widespread adoption of safe and reliable autonomous vehicles in the future.

## **1.2 Why Continual Learning in Automated Driving?**

As it can be seen, the perception system in an automated driving system will require several updates during its lifetime due to errors in the system as well as to extend the system's capabilities to new domains or new objects. The most secure and safe way to update such a system without risking catastrophic forgetting would be to optimize it on the entirety of the relevant data when new data is gathered. However, training the model on the entirety of the data as soon as new data is gathered is not feasible in many situations.

Firstly, offline training demands extensive computational resources. Tesla’s Autopilot system, for instance, requires a staggering 70,000 GPU hours for each re-training on their entire dataset [Kar20]. This immense computational requirement considerably increases the time until an update to the system is available, which in the worst case could lead to increased unavailability or downtime of the automated driving function, as safety concerns cannot be addressed in a timely manner. This enormous computational requirement also induces significant costs for updating such a system. A single training session of Tesla’s *Autopilot* on Azure, a widely used cloud computing platform, would amount to approximately 255,000 USD [Kar20]<sup>1</sup>.

Finally, offline training requires that the entirety of the training data is collectively available, which often cannot be guaranteed due to data privacy restrictions or data ownership. For instance, the European General Data Protection Regulation (GDPR) or Personal Information Protection Law (PIPL) of the People’s Republic of China strictly regulate outbound data transfers. Additionally, in some instances, perception systems may be supplied with pretrained models but lack the accompanying data required for adaptation or re-training, which will be especially the case for the so called *foundation models*, which are trained on billions of images that only companies like Google or Meta have access to.

Furthermore, the volume of data generated by automated driving systems poses a significant challenge in terms of storage capacity. A single test vehicle can produce around 19 terabytes of data per hour if all sensors are recorded [Göt21]. Even when data is carefully selected for training using active learning or corner case detection approaches, it is a severe challenge to save all the data. However, a silver lining is that while datasets are big, the resulting models trained on those datasets are small, for example the training dataset of the state-of-the-art model Segment-Anything-Model [Kir23] is 10TB, the weights of the model are only 2GB.

---

<sup>1</sup> Pricing is taken from <https://azure.microsoft.com/en-us/pricing/details/machine-learning/> for an ND96asr A100 v4 instance that costs 27.197 USD/hour for 8x Nvidia A100 GPUs.

Therefore, in the future when it would not be feasible to store all data in the same place or have access to the data at the same time, with continual learning it would be possible to condense the training data into a neural network so that it would not be required to have all data collectively available. At the same time continual learning will significantly reduce the training time at each model iteration.

## 1.3 Contributions

The work presented in this thesis makes the following contributions to the field of continual learning for semantic segmentation:

- A thorough empirical evaluation of the main categories of continual learning approaches in class- and domain-incremental learning is conducted for the first time for semantic segmentation and compared to similar investigations for classification. Therefore, two incremental benchmarks based on the automotive datasets Cityscapes and BDD100k are proposed that cover domain- and class-incremental learning settings. Relevant continual learning approaches were evaluated on those benchmarks, which revealed several distinct properties of how catastrophic forgetting manifests itself in continual semantic segmentation. Furthermore, replay-based learning is first proposed for class-incremental semantic segmentation to overcome inter-task confusion for classes that do not reappear in images of subsequent tasks. This work was published in the 2021 IEEE Intelligent Vehicles Symposium (IV 2021) [Kal21].
- A set of tools to measure, locate and interpret catastrophic forgetting in deep neural networks is constructed based on methods to measure the similarity of activations or weights as well as re-training and re-evaluation methods. These methods are compared in their ability to measure the effects of catastrophic forgetting to understand what inferences can be made when using them and when they potentially lead

to wrong conclusions. Additionally, this chapter introduces a new approach named decoder retraining to measure the impact of inter-task confusion on the encoder and decoder in class-incremental learning and firstly utilizes the stitching framework proposed by Csiszárík et al. [Csi21] for measuring the activation drift in a continual learning setting. These results were published partly in the 16<sup>th</sup> Asian Conference on Computer Vision (ACCV2022) [Kal22a] and partly in the Proceedings of the 2022 Joint Workshop of Fraunhofer IOSB and Institute for Anthropomatics, Vision and Fusion Laboratory [Kal23a].

- An analysis into the causes of catastrophic forgetting for class-incremental learning, in which the impact of the semantic shift of the background class is evaluated in three different task protocols: *overlapped*, *disjoint* and a novel *full disjoint* setup. The degree of activation drift in different layers is analyzed by stitching them with the network from the previous task, uncovering that the main cause of the catastrophic drop in performance in class-incremental semantic segmentation is the semantic shift of the background class. Forgetting mainly happens in the decoder layers of the model, where discriminative features for old classes of the encoder are assigned to new visually similar classes or to the background class. However, the re-appearance of previous classes in the background of subsequent training tasks also reduces the internal activation drift in the encoder. The results indicate that methods that do not use any form of replay fail to learn discriminative features for all classes. Specifically, the model is not able to distinguish old classes from new classes that are visually closely related, e.g. the classes *train* and *bus*. These results were published in the 16<sup>th</sup> Asian Conference on Computer Vision (ACCV 2022) [Kal22a].
- An analysis of the causes of catastrophic forgetting for domain-incremental learning and the role of feature reuse to overcome the severe activation drift. This analysis reveals that the major cause of forgetting in domain-incremental learning is the shift of low-level representations in the first convolution layer, which adversely affects the population statistics of the following batch normalization layer. Using

different augmentation strategies to match the target domains in color statistics and in the frequency domain, the experiments demonstrate that color-invariant features stabilize the features in the early layers, as they are not changed when the model is adapted to a new domain. With a combination of pre-training, augmentations and exchanged normalization layers an overall reduction of forgetting by around 20 mIoU is achieved compared to fine-tuning without using any form of replay. These results highlight that pre-training and augmentations are often overlooked ingredients for continual learning. The findings were published in the IEEE / CVF Computer Vision and Pattern Recognition Conference 2023 (CVPR 2023) [Kal23c].

- An analysis of various architectural decisions that were introduced in recent years for segmentation models for continual learning, in which different encoder and decoder architectures and normalization layers are evaluated in class- and domain-incremental semantic segmentation. The experiments highlight that the neural architecture influences both the extent and the location in the network where the model is affected by catastrophic forgetting. Thereby, the architecture of the model has a significant impact on the model's plasticity and stability. However, the findings suggest that the increased robustness towards catastrophic forgetting of modern architectures such as ConvNeXt and Vision Transformers does not originate only in the self-attention layer but is also linked to other micro and macro design choices. This work was published in the 2023 IEEE Intelligent Vehicles Symposium (IV 2023) [Kal23b].
- As samples for replay-based learning are mostly selected at random, it is observed that the performance can vary significantly compared to other continual learning approaches. Therefore, several sample selection strategies for replay-based continual semantic segmentation were developed and adapted, which achieve more stable results compared to random selection. Effective replay strategies aid in stabilizing feature representations, particularly in deeper network layers, during domain-incremental training. It is shown that approximating the distribution

of internal task data representations or selecting samples with median entropy yields the best results and that in class-incremental learning, class-balancing is the most crucial factor for sample selection. This work was published in the 2022 IEEE Conference on Intelligent Transportation Systems (ITSC 2022) [Kal22b].

## 1.4 Thesis Outline

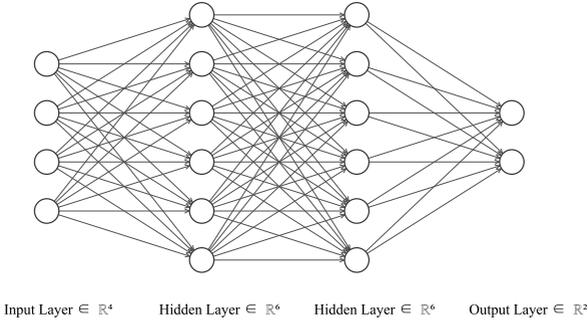
The goal of this thesis is to investigate the causes and effects of catastrophic forgetting in continual semantic segmentation, answering how, where and why catastrophic forgetting arises in deep neural networks. Therefore, Chs. 2 and 3 introduce the fundamentals of deep learning in the context of continual learning and discuss the recent advances in continual learning and semantic segmentation. Next, Ch. 4 studies how existing continual learning approaches that were developed and evaluated for classification tasks perform and behave in class- and domain-incremental semantic segmentation. The results open up several research questions, of how catastrophic forgetting is affecting models differently for semantic segmentation than for classification and why and how catastrophic forgetting affects the segmentation model in class- and domain-incremental learning. To answer these question, Ch. 5 presents and evaluates several measures and methods that allow deeper insights into the internal activation and weight changes of the segmentation model in continual learning. In Ch. 6, the main part of this thesis, these tools are utilized to investigate the principles of catastrophic forgetting for class-incremental learning (Sec. 6.1) and domain-incremental semantic segmentation (Sec. 6.2). Finally, Ch. 7 examines how different neural architecture choices affect catastrophic forgetting and why vision transformers (VTs) are more robust against catastrophic forgetting.

## 2 Background

The following chapter establishes the required theoretical basis on which approaches for continual learning for semantic segmentation are built. The first part provides a brief introduction to artificial neural networks as well as convolutional neural networks and vision transformers, which are essential in understanding the challenges that arise in continual learning. Section 2.3 first introduces the concept of continual learning and its desiderata, and then explains the challenges that arise, such as catastrophic forgetting and its underlying effects. The final section of this chapter will present metrics that will be used to evaluate methods for continual semantic segmentation throughout this thesis. Parts of the sections regarding continual learning and artificial neural networks are based on the author's publication in [Kal23a].

### 2.1 Artificial Neural Networks

Artificial neural networks (ANNs) are a class of machine learning models inspired by biological neural networks in the brain. ANNs are capable of learning complex mappings between input and output data, making them powerful tools for a wide range of applications, including image and speech recognition, natural language processing and autonomous driving. A typical ANN consists of multiple layers of neurons. Figure 2.1 depicts a basic neural network with an input, an output and two hidden layers.



**Figure 2.1:** Artificial Neural Network with two hidden layers.

In typical feed-forward networks neurons are only connected with neurons of subsequent layers. Overall, an ANN model  $f$  consists of  $N$  consecutive layers of neurons  $l_n$  so that  $f = l_N \circ \dots \circ l_1$ , where  $l_n : \mathcal{A}_{n-1} \rightarrow \mathcal{A}_n$  are mappings between the activation spaces  $\mathcal{A}_{n-1}$  and  $\mathcal{A}_n$  with  $\mathcal{A}_0 = \mathcal{X}$ . Generally, the goal of an ANN is to approximate mapping function  $f^* : \mathcal{X} \mapsto \mathcal{Y}$  which maps an input space  $\mathcal{X}$  to target output space  $\mathcal{Y}$ . To approximate function  $f^*$  the parameters  $\theta$  of the model  $f$  are optimized w.r.t to a loss function  $\mathcal{L}$  that measures the agreement between the target output  $y$  and the model's output  $f(x) = \hat{y}$ .

Given a learning task  $T = \{(x_m, y_m)\}_{m=1}^M$  that consists of a set of  $M$  inputs  $x_m \in \mathcal{X}$  and corresponding labels  $y_m \in \mathcal{Y}$ , the model is optimized by taking steps in the direction of the negative gradient of the empirical risk  $g$  over  $T$  w.r.t to the loss  $\mathcal{L}$ . Most commonly  $g$  is approximated by calculating the stochastic gradient  $\tilde{g}$  on a mini batch  $T' \subset T$  with  $p_i$  as the sampling probability for a training sample.

$$\tilde{g} = \sum_{(x_i, y_i) \in T'} p_i \nabla \mathcal{L}(f(x_i), y_i) \quad (2.1)$$

Typically  $p_i$  is uniform for all training samples, so the expectation of  $\tilde{g}$  is equal to  $g$ . The gradients for each individual weight  $\theta$  of the model are computed with the backpropagation algorithm [Rum86]. During optimization,

the optimizer takes gradient steps on randomly selected mini batches  $T'$  until convergence. Once a local minimum is found, the neural network is put into production and utilized to make predictions for new unseen data.

## 2.2 Deep Neural Networks for Computer Vision

The universal approximation theorem [Cyb89, Hor91] states that ANNs are universal function approximators, meaning that they can approximate any continuous function to arbitrary accuracy given a bounded number of hidden layers and an arbitrary number of neurons in each layer. Similarly, Lu et al. [Lu17] proved the arbitrary depth-case given a bounded width of the neural network for various continuous functions. They conclude that “The width and the depth are two key components in the design of a neural network architecture. Width and depth are both important and should be carefully tuned together for the best performance of neural networks, since the depth may determine the abstraction level but the width may influence the loss of information in the forwarding pass.”

That specifically entails that one way to improve the performance of a model is to increase the number of hidden layers of the ANN. Generally, models with more than one hidden layer are termed deep neural networks (DNNs). The wide success of DNNs in computer vision was achieved with the introduction of convolutional neural networks (CNNs) and more recently with the introduction of the vision transformer (VT), because of their specific inductive biases that the architecture provides for vision tasks. In the following, the main ideas about these architectures are explained.

### 2.2.1 Convolutional Neural Networks

CNNs are a special type of ANNs that were specifically designed to optimize the performance for processing digital images and videos. The human visual cortex served as inspiration for their design. Neurons in this section of the

brain only respond to stimuli inside their immediate region, known as their receptive field. The fundamental distinction between ANNs and CNNs is the usage of *convolutional layers*, which reduce the number of parameters for the ANN. This is achieved with two key ideas:

- 1 Local Receptive Fields: CNNs assume that elements that are close to a region of interest in the input data are more likely to be related and thus more relevant than elements that are more distant. Convolutional layers capture this by connecting neurons of a layer only to a local part of the neurons of the previous layer to extract only features from a fixed local neighborhood. Thereby, reducing the number of connections between neurons.
- 2 Weight Sharing: CNNs assume that the same features or patterns can appear in different parts of the input. Therefore, the same set of weights is used across multiple spatial locations of the input. By sharing weights, the network learns to detect and recognize features irrespective of their specific position, leading to translational invariance.

These design choices introduce inductive biases that not only reduce the amount of parameters, but also lead to better generalization capabilities as CNNs become, in part, translation invariant.

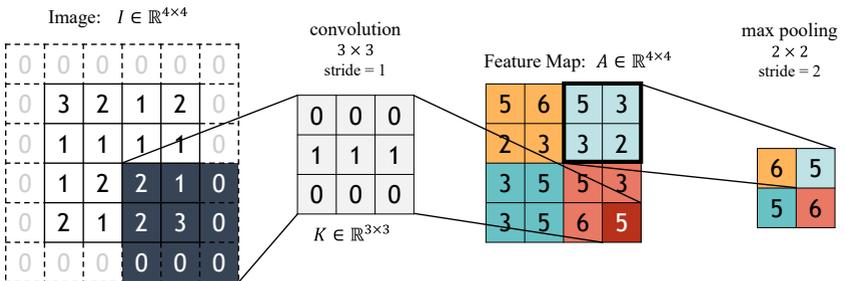
The first CNNs consisted only of very few layers and were capable of classifying hand-written digits [LeC89, Fuk80]. With the release of large-scale datasets such as ImageNet and efficient computation on Graphics Processing Units (GPUs), CNNs outperformed classical computer vision approaches for the first time. Since then, significant improvements have been made, allowing the models to be stacked even deeper [He16] and to converge faster [Iof15], which has led to the wide adoption of CNNs in several computer vision tasks such as object detection, semantic segmentation, visual question answering, image editing and image generation.

The main component of a CNN is the convolutional layer, which computes the activation of each neuron with discrete convolution. The input of a neuron in the convolutional layer is calculated as the inner product of a filter kernel

$K \in \mathbb{R}^{w_k \times h_k}$  with the currently underlying image section. Given a filter kernel  $K$  with the size  $w_k \times h_k$  and an input image  $I$  with the size  $w \times h \times r$  the output  $S(x,y)$  at a position  $(x,y)$  is computed as:

$$S(x,y) = \sum_{i=1}^{w_k} \sum_{j=1}^{h_k} I(x-i, y-j)K(i, j) \quad (2.2)$$

where  $K(i, j)$  represents one element of the given kernel. The output size  $w_{\text{out}} \times h_{\text{out}}$  is directly affected by the padding and stride of the convolution. The stride specifies the distance between the spatial locations where the filter kernels are applied. That means with a stride of one, the width and height of the output will be equal to the input width and height, which is illustrated in Fig. 2.2. Typically, a single convolutional layer consists of multiple filter kernels that operate on the same input. Each filter applies a convolution operation to the input, extracting relevant features. The resulting features are concatenated and result in a feature map  $\mathbf{A}$  of the size  $w_{\text{out}} \times h_{\text{out}} \times l$  where  $l$  denotes the number of filter kernels.



**Figure 2.2:** Illustration of the convolution and a sequential max pooling operation. To ensure that the output feature map has the same dimensions as the input, the image is first padded with zeros. A filter kernel of size  $3 \times 3$  is convolved with an input image of size  $4 \times 4$ . The resulting output feature map is then max-pooled with a kernel of size  $2 \times 2$  and a stride of 2.

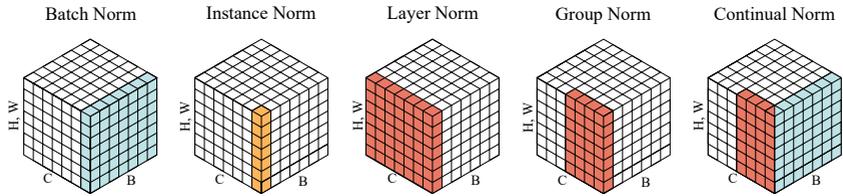
After a convolutional layer, a *pooling layer* is typically added to reduce the size of feature maps and improve computation speed. A pooling operation is

defined by the filter size, usually  $(2 \times 2)$  or  $(3 \times 3)$ , the stride and the pooling function. The most commonly used pooling function is *max pooling*, which returns the maximum value within the receptive field of the filter. In order to reduce the size of the feature maps, pooling operations are mostly used with stride 2, which reduces the size of the feature map to a quarter of its original size, as illustrated in Fig. 2.2.

Another essential component of CNNs are *normalization layers* that enable faster convergence during training by stabilizing the distribution of input values within the network, making the layers more robust to changes in the input statistics [Iof15]. Given a mini-batch of feature maps  $a = (\mathbf{A}_1, \dots, \mathbf{A}_B)$  of size  $B$ , normalization layers normalize the input as:

$$a' = \gamma \left( \frac{a - \mu}{\sqrt{\sigma^2 + \epsilon}} \right) + \beta \quad (2.3)$$

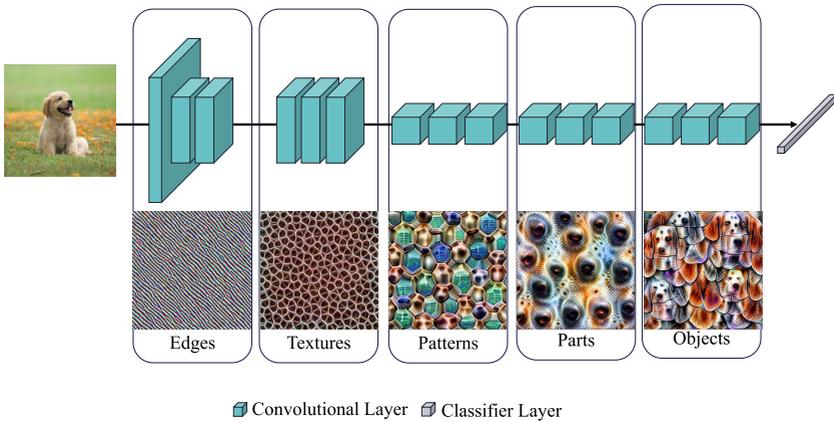
where  $\mu$  and  $\sigma^2$  are the mean and variance of the input features, respectively. The learnable parameters  $\gamma, \beta$  are affine transformations to retain the representational capacity of the layer after normalization. There are several types of normalization layers that normalize the activations along different dimensions of the feature map, as illustrated in Fig. 2.3. The most commonly used normalization layer is batch normalization (BN), which normalizes the input across the batch dimension. To achieve deterministic behavior during inference that is independent of other samples within a batch, the mini-batch variance and mean are replaced with the global population mean and variance of the training dataset, which are computed using an exponential running average during training.



**Figure 2.3:** Illustration of different normalization methods derived from [Wu18b] and [Pha22]. Each big cube represents a feature map tensor, with  $B$  as the batch axis,  $C$  as the channel axis, and  $(H, W)$  as the spatial axis. The pixels in blue are normalized by the same moments across the batch dimension. Pixels in red are normalized by the same moments calculated within the sample and orange pixels are only normalized along the spatial dimensions.

Other commonly used normalization layers are layer normalization (LN) that normalizes across the spatial dimensions and the channel dimensions, instance normalization (IN) that normalizes only along the spatial dimensions, group normalization (GN) [Wu18b] that normalizes along spatial dimensions and specific grouped channels and continual normalization (CN) which combines group and batch normalization.

With these main components, most CNNs follow a similar structure with blocks of convolutions, normalization and pooling layers and residual connections that enable CNNs to improve their classification accuracy with increased depth [He16]. With this hierarchy, CNNs naturally extract features with increasing levels of abstraction through the succession of convolutions. Figure 2.4 shows that early layers respond to edges, then later layers capture textures and even deeper layers respond to specific patterns or parts of objects and layers close to the output correspond to entire objects [Zei14]. The hierarchical nature of those features of a CNN will play a role in understanding the effects of catastrophic forgetting in Ch. 6.



**Figure 2.4:** Feature Visualization from different layers of a CNN trained on ImageNet. The feature visualizations are taken from Olah et al. [Ola17]

## 2.2.2 Vision Transformers

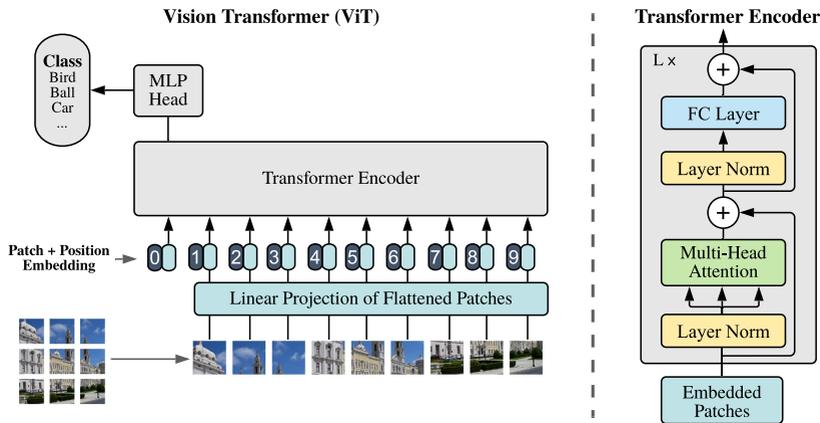
The VT is a neural architecture that was proposed as an alternative to CNNs, which have been the dominant approach for many computer vision tasks for years. The first VT that was widely adopted is ViT [Dos21]. The ViT model is based on the transformer architecture, which was originally introduced for natural language processing tasks. The transformer architecture uses the *self-attention* block to process sequential data, such as sentences. Self-attention enables the model to weigh the importance of different parts of an input sequence so that it is able to draw global dependencies between input and output [Vas17]. To achieve this, every element in the input sequence is first transformed to a high-dimensional vector named *token*. Then for each token a query vector  $Q$ , a key vector  $K$  and a value vector  $V$  is computed using the corresponding learned weight matrices:  $W_q$ ,  $W_k$  and  $W_v$ . These vectors are used

to compute a weighted representation of the input sequence  $X$  of length  $N$ :

$$Q = XW_q, \quad K = XW_k, \quad V = XW_v \quad (2.4)$$

$$\text{SelfAttention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (2.5)$$

where  $d_k$  is the dimension of the key vectors. ViT uses the exact same self-attention mechanism, but in order to work on image data, each image is first divided into a set of equally sized patches, which are then used as a sequence of input tokens [Dos21]. The architecture is shown in Fig. 2.5. The main advantage of VT over CNNs is the ability to capture long-range dependencies between different regions of the image. A downside of this property is that they lack some of the inductive biases of CNNs such as translation equivariance and locality. That is why they often require more training data to generalize well [Dos21]. However, it was discovered that when VTs are trained on a sufficiently large amount of training data, they learn convolutional-like configurations in early layers of the network [Cor19, Rag21]. Therefore, hybrid models were proposed that use the flexibility of the self-attention mechanism to capture long-range dependencies in the image but also re-introduce the inductive biases of CNNs [dAs21, Dai21].



**Figure 2.5:** Vision Transformer architecture: The image is divided into equally sized patches and embedded into a high dimensional vector space using a convolutional layer with stride equal to the patch size. To each resulting patch a positional encoding is added. Each transformer block (displayed on the right) consist of two layer normalization layers, a multi-head self-attention layer and a fully connected layer. Image from Dosovitskiy et al. [Dos20]

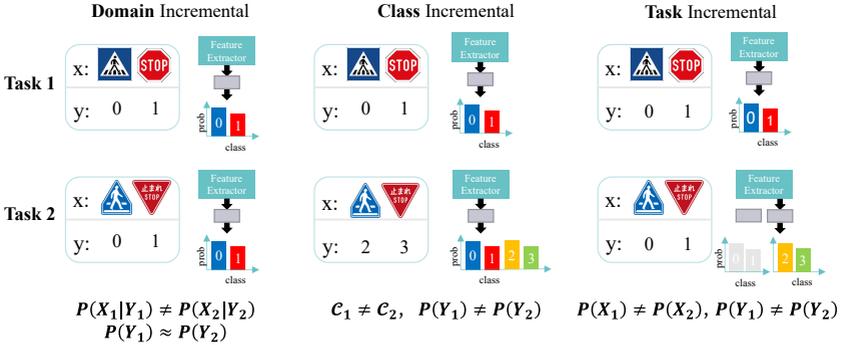
## 2.3 Continual Learning

Continual learning, also referred to as life-long learning, is a sub-field of machine learning that focuses on enabling machine learning systems that have the ability to learn from a continuous stream of data while gradually extending the acquired knowledge without negatively interfering with previously learned knowledge [Del21]. This is in contrast to the common practice for developing machine learning models, where it is generally assumed that all required data is collectively available during training, meaning that the data distribution stays fixed during the life-cycle of the machine learning system. Therefore, instead of learning from a single task  $T$  with a fixed training distribution, in continual learning the model is optimized on a sequence of tasks  $T_{1..K}$ , while only having access to data from one task at a time. Each task can introduce new classes or visually distinct instances of

the same classes [Hsu18]. Generally, given marginal probability distributions over inputs  $P(X_k)$  and outputs  $P(Y_k)$  and a set of classes  $\mathcal{C}_k$  for a task  $T_k$ , three different scenarios are commonly defined for continual learning [Van18, Hsu18, Ven22]:

- Class-incremental learning, where each new task adds a set of novel classes  $\mathcal{S}_k$  to the existing set of classes  $\mathcal{C}_k$ . So that  $\mathcal{C}_2 = \mathcal{C}_1 \cup \mathcal{S}_2$ .
- Domain-incremental learning, where the classes remain the same but the images of each task are obtained from distinct distributions and thus have distinct visual appearances. In this setting  $P(X_k|Y)$  is changing so that  $P(X_1|Y_1) \neq P(X_2|Y_2)$  while  $P(Y_1) \approx P(Y_2)$  and  $\mathcal{C}_2 = \mathcal{C}_1$ .
- Task incremental learning is similar to class-incremental learning, with the exception that for each sample a task identifier is given, so that the model does not need to discriminate between classes of different tasks. This setting reduces the complexity of the incremental learning problem. This setting is relatively unlikely since a sufficiently reliable task identifier is generally difficult to establish in most cases; hence, this thesis will not focus on this type of continual learning.

A visualization of these settings is shown in Fig. 2.6. In the domain-incremental setting (left), the classes stay the same, but the visual appearance of the classes is changing. In the class-incremental setting (middle), the differently looking street signs are introduced as new classes that have to be distinguished from the previously learned classes. On the right, in the task incremental setting, the new classes are classified independently from the old classes as the output spaces between different tasks are disjoint. Therefore, in this scenario the model is not required to distinguish between classes of  $T_1$  and  $T_2$ .



**Figure 2.6:** The most commonly defined incremental learning scenarios as defined by [Hsu18]. In each sub-figure the boxes represent the input for training,  $x$  denotes the input and  $y$  the corresponding labels.

### 2.3.1 Catastrophic Forgetting

A major challenge in continual learning is to overcome catastrophic forgetting, where the model overwrites knowledge learned from previous tasks while learning a new task [Fre94, McC89]. Catastrophic forgetting arises because the model cannot be jointly optimized on all available data, so during optimization on a task  $T_k$  data from task  $T_{k-1}$  is not available. This changes the stochastic optimization procedure introduced in Eq. (2.1), because the sampling probability  $p_i$  is no longer uniform for all training samples but instead only uniformly distributed over samples belonging to the current task  $T_k$ . Furthermore, as training data from previous tasks is no longer available during optimization on task  $T_k$ , the sample distribution for all samples of previous tasks  $\{T_t | t \neq k\}$  is  $p_i = 0$ . This changes the gradient calculation for optimization up to the current task  $T_k$  for a sequence of tasks  $T_{1..K}$  from:

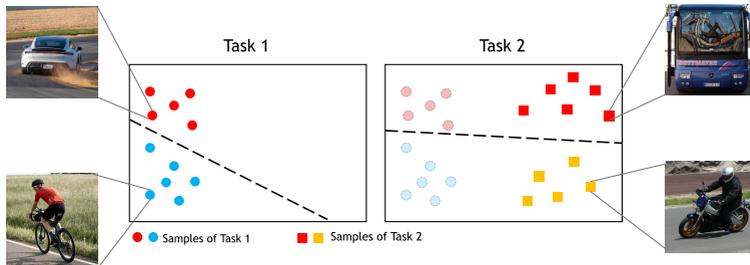
$$\tilde{g} = \sum_t^K \sum_{(x_i, y_i) \in T_t} p_i \nabla \mathcal{L}(f(x_i), y_i) \quad (2.6)$$

to a sequence of distinct optimization processes that start with the model trained on the previous task  $f_{k-1}$ :

$$\tilde{g}_k = \sum_{(x_i, y_i) \in T_k} p_i \nabla \mathcal{L}(f(x_i), y_i) \quad (2.7)$$

Thus, when training on task  $T_k$  the optimization process disregards other tasks' distributions, except for the initialization as  $f_{k-1}$ , in the sense that the gradients are computed only based on the loss on the current task dataset. This is the root cause of catastrophic forgetting in continual learning. The effects this optimization process has on neural networks are not fully understood; however, Masana et al. [Mas20] have identified four effects of how catastrophic forgetting manifests itself in continual learning.

- **Weight Drift:** During optimization on  $T_k$ , the weights of the model that was trained on  $T_{k-1}$  are updated without regard to the previous task. This can lead to unwanted changes to weights that were important for the previous task, which results in a drop in performance on task  $T_{k-1}$ .
- **Activation Drift:** A change in the weights of the model directly results in a change in the internal activations of the neurons and the output of the model. While activation drift is a direct result of weight drift, activation drift additionally takes the input data distribution into account.
- **Inter-task confusion:** The objective of class-incremental learning is to correctly discriminate between classes from all tasks. However, as the classes are never jointly trained, the learned features are not optimized to discriminate classes from different tasks, as shown in Fig. 2.7. Inter-task confusion is also linked to task-specific spurious features that can emerge in domain-incremental learning [Les22].
- **Task-recency bias:** In the class-incremental setting, the model is optimized to predict new classes without considering the old classes. This results in an increased bias for the most recently learned classes, that is evident in the bias values of the classification layers.



**Figure 2.7:** Visualization of inter-task confusion in class-incremental learning. As classes of Task 1 (*car* and *bicycle*) and classes of Task 2 (*bus* and *motorcycle*) are never trained at the same time, the classifier never learns to discriminate between bicycle and motorcycle, which causes inter-task confusion.

### 2.3.2 Desiderata

Continual learning is currently studied for many existing machine learning tasks such as computer vision, natural language processing and reinforcement learning. However, to keep focus, the scope of this work will be limited to the widely adopted computer vision tasks of classification and semantic segmentation. The overarching goal in continual learning for a model is to learn from an infinite stream of data and to gradually acquire new knowledge without negatively interfering with previously learned knowledge. According to De Lange et al. [De 19] an ideal continual learning algorithm should meet the following 10 desiderata, stated verbatim:

- 1 Constant memory: Regardless of the number of tasks or the length of the data stream, the continuous learning algorithm should consume constant memory. The reason for this is to avoid dealing with unbounded systems.
- 2 No task boundaries: Data does not arrive in incremental batches but in a continuous stream, where clear task boundaries are not defined, e.g. the transition between data from different domains is fluent. Having the ability to learn from input data without having to split it up into specific tasks

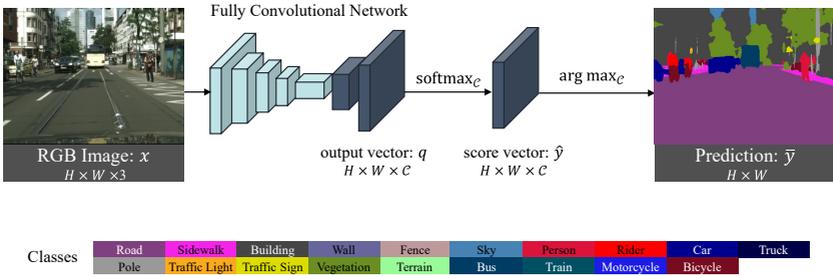
lends a great deal of flexibility to the continuous learning process.

- 3 Online learning: The ability to learn from a continuous stream of data without offline training of large batches or separate tasks.
- 4 Forward transfer: This characteristic indicates the importance of the previously acquired knowledge to aid the learning of new tasks.
- 5 Backward transfer: A continual learning system should not only aim at retaining previous knowledge but preferably also at improving the performance on previous tasks when learning future related tasks.
- 6 Problem agnostic: A continual learning method should be general and not limited to a specific setting (e.g. only classification).
- 7 Adaptive: Being able to learn from unlabeled data would increase the method applicability to cases where original training data no longer exists and further open the door to a specific user setting adaptation.
- 8 No test time oracle: Continual learning methods should not be dependent on task oracles to make predictions.
- 9 Task revisiting: When revisiting a previous task again, the system should be able to successfully incorporate the new task knowledge.
- 10 Graceful forgetting: The selective forgetting of unimportant information is an essential mechanism to maintain stability and plasticity in an bounded system.

Thus far there is no common understanding on the importance of each of these desiderata, so proposed algorithms for continual learning only adhere to a varying selection of the desiderata, respective of what their specific purpose

is. Therefore, many of these desiderata are better understood than others. Especially, the property of continual learning approaches being task agnostic is mostly overlooked, as proposed algorithms are only ever evaluated either on object detection, classification or semantic segmentation. This also explains why state-of-the-art approaches for continual learning in semantic segmentation, classification and other tasks are different.

## 2.4 Semantic Segmentation



**Figure 2.8:** Semantic segmentation: Given an RGB image  $x$  the CNN predicts a score vector  $\hat{y}$ . The model is optimized by calculating the cross-entropy loss between the ground-truth  $y$  and score vector  $\hat{y}$ .

The task of semantic segmentation is to assign a semantic class, out of a set of pre-defined classes  $\mathcal{C}$ , to each pixel in a given image. A training task  $T = \{(x_m, y_m)\}_{m=1}^M$  consists of a set of  $m$  images  $x \in \mathcal{X}$  with  $\mathcal{X} = \mathbb{R}^{H \times W \times 3}$  and corresponding labels  $y \in \mathcal{Y}$  with  $\mathcal{Y} = \mathcal{C}^{H \times W}$ . Given the task  $T$  the goal is to learn a mapping  $f : \mathcal{X} \mapsto \mathbb{R}^{H \times W \times |\mathcal{C}|}$  from the image space  $\mathcal{X}$  to a vector  $q$  that is normalized using the softmax function to obtain the score vector  $\hat{y}$ . The score vector  $\hat{y}$  gives a score for each pixel  $i \in I$  with  $I = \{1, \dots, H\} \times \{1, \dots, W\}$  in the input image  $x$  and each class of  $\mathcal{C}$  that the pixel  $x_i$  belongs to class  $c \in \mathcal{C}$ . The final segmentation mask is then computed as  $\bar{y}_i = \arg \max_{c \in \mathcal{C}} \hat{y}_{i,c}$ . The model  $f$  is trained by optimizing the cross-entropy between the estimated

score vectors  $\hat{y}$  and the one-hot encoded ground-truth class labels  $y$ .

$$\mathcal{L}_{ce}(y, \hat{y}) = -\frac{1}{H \cdot W} \sum_{i \in I} \sum_{c \in C} y_{i,c} \log(\hat{y}_{i,c}) \quad (2.8)$$

Semantic segmentation has seen significant progress in recent years, due to large-scale datasets as well as the use of CNNs for this task. The first breakthrough in this field came with the introduction of fully convolutional networks (FCN) [Lon15]. Subsequent seminal works made several improvements to the architecture, such as using skip-connections [Ron15], spatial pyramid pooling [Zha17], dilated convolutions [Che17a] and using vision transformers [Tou21, Xie21]. Most recently, MaskFormer [Che21b, Che22] was introduced, which converts the per-pixel classification into mask classification. Instead of predicting a class probability for each pixel, as in previous semantic segmentation works, MaskFormer predicts a set of binary masks, each of which corresponds to a single class prediction. In this thesis, the widely used DeepLabV3+ [Che18] and ERFNet [Rom18] will be mainly used, as DeepLabV3+ is the most common choice for evaluating methods for continual semantic segmentation and ERFNet as it is a commonly used real-time capable method for automotive applications.

## 2.5 Evaluation Metrics

In the following, the quantitative evaluation metrics used throughout this thesis are presented. The first section introduces the most commonly used semantic segmentation metric mean intersection over union (mIoU) and discusses its benefits in comparison to other metrics. Next, specific metrics are presented that are used to compare the performance of different continual learning methods.

### 2.5.1 Semantic Segmentation

The evaluation of the performance of different semantic segmentation models requires quantitative metrics to measure the accuracy of segmentation results. The most common metrics used in semantic segmentation are: pixel accuracy (PA), mean pixel accuracy (mPA), intersection over union (IoU) and mean intersection over union (mIoU) [Lon15]. Generally, for a binary classification task there are four outcomes for a prediction:

- *true positive* (TP): The model predicts a *positive* class label, and the ground truth label is also *positive*. In the context of semantic segmentation, true positives occur when the model assigns the correct label to a pixel that belongs to the object class.
- *false positive* (FP): The model predicts a *positive* class label, but the ground truth label is *negative*. In the context of semantic segmentation, false positives occur when the model assigns an incorrect label to a pixel that does not belong to the true object class.
- *true negative* (TN): The model predicts a *negative* class label, and the ground truth label is also *negative*. In the context of semantic segmentation, true negatives occur when the model correctly assigns a background label to a pixel that does not belong to the object class.
- *false negative* (FN): The model predicts a *negative* class label, but the ground truth label is *positive*. In the context of semantic segmentation, false negatives occur when the model fails to assign the correct object label to a pixel that belongs to the object class.

Given the number of pixels of class  $i$  predicted to belong to class  $j$  as  $n_{i,j}$  and  $t_i$  as the total number of pixels of class  $i$  and  $C$  as the number of classes,  $n_{j,j}$  denotes the number of *true positives* of class  $j$  and  $t_j$  the total number of pixel labeled as class  $j$ .

### Pixel Accuracy

The pixel accuracy measures the ratio between correctly classified pixels and the total number of pixels. A downside of this simple metric is that it does not account for the class imbalance in an image and can be misleading when the number of pixels for classes varies significantly, e.g. if the class *street* took up 90 % of the image, a model that predicts everything as street would achieve 90 % accuracy. Therefore, mean pixel accuracy (mPA) calculates the accuracy for each class separately and then takes their mean. This metric provides a more balanced view of the model's performance, but it does not directly account for false positive predictions.

$$\text{PA} := \frac{\sum_{j=1}^C n_{j,j}}{\sum_{j=1}^C t_j} \quad \text{mPA} := \frac{1}{C} \sum_{j=1}^C \frac{n_{j,j}}{t_j} \quad (2.9)$$

### Mean Intersection over Union (mIoU)

Mean intersection over union is the most popular metric for measuring the performance of semantic segmentation models and is equivalent to the Jaccard Index. It calculates the intersection over union (IoU) for each class separately and then takes their mean. The IoU<sub>j</sub> for a class *j* is calculated by dividing the number of *true positive* predictions  $n_{j,j}$  by the union of *true positive*, *false positive* and *false negative* pixels.

$$\text{IoU}_j := \frac{n_{j,j}}{n_{j,j} + \sum_{i=1}^C (n_{j,i} + n_{i,j})} \quad i \neq j \quad (2.10)$$

Most commonly instead of providing the IoU for each individual class, the mean is reported as:

$$\text{mIoU} := \frac{1}{C} \sum_{j=1}^C \text{IoU}_j \quad (2.11)$$

## 2.5.2 Continual Learning Metrics

The performance of a model in continual learning is usually evaluated from two perspectives: its ability to learn new tasks and its ability to retain knowledge of previous tasks. To assess this, the model’s performance is evaluated on all previous and future test sets after each training task. In the context of semantic segmentation, the mIoU of the model trained on all tasks up to task  $T_k$  and evaluated on task  $T_v$  is denoted as  $\text{mIoU}_{k,v}$ . Hence, the performance of a model trained on task  $k = 0$  and evaluated on  $v = 1$  is denoted as  $\text{mIoU}_{0,1}$ . Mirzadeh et al. [Mir22b] defines the following metrics:

- *average forgetting*: measures the drop of accuracy between the peak mIoU on a test set during the training sequence and the mIoU achieved after training on all tasks. It is used to measure the stability of a model. For a task sequence of length  $K$  it is calculated as:

$$\frac{1}{K-1} \sum_{v=1}^{K-1} \max_{k \in \{1, \dots, K-1\}} (\text{mIoU}_{k,v} - \text{mIoU}_{K,v}) \quad (2.12)$$

- *average accuracy*: measures the average accuracy after the model has been trained on a task sequence of length  $K$ . This is the most common metric that has been used to compare the final performance of a continual learning algorithm. It is computed as:

$$\frac{1}{K} \sum_{v=1}^K \text{mIoU}_{K,v} \quad (2.13)$$

- *learning accuracy*: measures the plasticity of a model, by averaging the accuracy for each task directly after it is learned.

$$\frac{1}{K} \sum_{k=1}^K \text{mIoU}_{k,v} \quad (2.14)$$

- *offline accuracy*: measures the accuracy a model achieves when it is trained on the data of all tasks together.

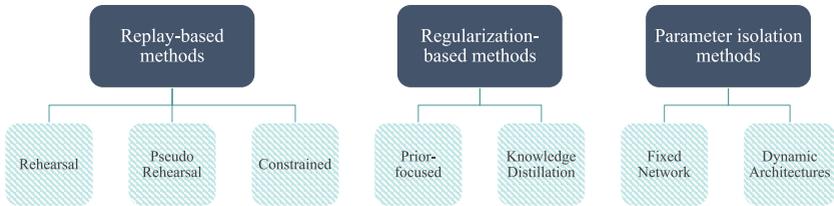


## 3 Related Work

The following chapter provides an overview of the relevant literature and research in the field of continual learning for semantic segmentation. In the first part, the most prominent strategies proposed in the literature for classification are presented and their strengths and weaknesses are discussed. The second part of the chapter focuses on the state-of-the-art approaches for continual semantic segmentation, with an emphasis on class-incremental learning and continual unsupervised domain adaptation. Finally, Sec. 3.3 discusses related work that has studied the effects of catastrophic forgetting in classification tasks.

### 3.1 Continual Learning Approaches

Approaches to mitigate the effects of catastrophic forgetting are commonly categorized into three categories [De 19]: methods that replay a selection of data from previous tasks, methods that regularize the model’s weights or outputs and methods that isolate specific parameters for each task. The main categories and their subcategories are shown in Fig. 3.1. In the following the main methods and their known limitations are discussed.



**Figure 3.1:** The main categorization of continual learning methods proposed by De Lange et al. [De 19].

### 3.1.1 Replay-Based Methods

Replay-based approaches store a selection of old training data for rehearsal during training on new data. Standard rehearsal methods store previous data either as raw image and label pairs [Reb17, Wu19a, Cas18] or internal representations of the model [Hay20, Ach20]. This replay buffer is constrained to a fixed number of training samples, so once the buffer is full, samples have to be removed before new samples can be added. Therefore, a sample selection strategy decides which samples should be stored in the replay buffer and which should be discarded. The most commonly used sampling strategies are herding [Wel09] and random sample selection for replay, which are reported to be very strong baselines [Alj19]. Sample strategies for class- and domain-incremental semantic segmentation will be discussed in more detail in Sec. 4.4.

Pseudo-rehearsal methods use Generative Adversarial Networks (GANs) or auto-encoders to approximate the previous training distribution so that pseudo-samples can be generated during training on a new task [Shi17, Zha19a, Wu18a]. However, the major drawback of these methods is that the GAN or auto-encoder is also affected by catastrophic forgetting, as the desideratum of a bounded system does not allow to train a new generator for each task [Zha19a]. Moreover, constrained optimization is another form of replay, in which parameter updates are constrained so that the loss for samples in the replay buffer is not increased [Lop17, Cha18b, Alj19]. However, constrained optimization is often outperformed by naive replay, as the

constraints can often be too restrictive, leading to overfitting on the memory buffer [Ver21].

Replay-based methods are reported to be the most effective methods to mitigate catastrophic forgetting in continual learning for image classification [Hsu18, De 19] and might be required for class-incremental learning to solve inter-task confusion [Ven20]. Verwimp et al. [Ver21] study why replay is so effective at mitigating forgetting even though the model overfits on the rehearsal memory. Their results suggest that the replay buffer prevents the continually trained model from leaving the first low-loss region during optimization on new data, possibly harming generalization.

### 3.1.2 Regularization-Based Methods

Regularization-based approaches can be divided into prior-based regularization approaches and knowledge distillation-based approaches. Prior regularization methods constrain parameter updates of parameters that were most crucial for solving previous tasks [Alj18, Kir15, Zen17]. Therefore, they add an additional regularization term to the loss function, which varies the weights' plasticity based on the estimated importance  $\Omega_j$  of the weights in previous tasks.

$$\mathcal{L}_{\text{reg}}(\tilde{y}, y, \theta) = \mathcal{L}_{\text{ce}}(y, \hat{y}) + \lambda \sum_j \Omega_j (\theta_j - \tilde{\theta}_j)^2 \quad (3.1)$$

where  $\tilde{\theta}_j$  are the old network parameters and  $\lambda$  is a hyper-parameter to balance the regularization. The main difference between methods in this category is in the way  $\Omega_j$  is calculated. Kirkpatrick et al. [Kir15] introduce Elastic Weight Consolidation (EWC), in which they estimate the importance  $\Omega_j$  for each parameter, using the diagonal of the Fisher information matrix. Zenke et al. [Zen17] propose to estimate  $\Omega_j$  by accumulating changes in each parameter along the entire learning trajectory. Memory Aware Synapses (MAS) [Alj18] computes the importance  $\Omega_j$  based on how sensitive the predicted output function is to a change in this parameter  $\theta_j$  for a new given sample. Other approaches combine the Fisher Matrix approximation and path

integral of Zenke et al. [Zen17] to calculate the importance of each parameter [Cha18a]. Recent regularization methods are mostly only evaluated in the task-incremental setting, where a task identifier is available at test time to select the corresponding classifier head [Alj18, Kir15, Cha18b]. In class-incremental learning, where such a task identifier is not available, regularization methods are outperformed by knowledge distillation-based approaches and replay [Mas20, Hsu18]. Prior-regularization methods are assumed to be limited as they cannot overcome inter-task confusion (Fig. 2.7) without additional replay samples [Van18, Les20].

The other class of regularization methods are based on the idea of knowledge distillation for neural networks [Hin15]. Knowledge distillation was first introduced to distill the knowledge from big or ensemble models, named *teacher*, to a smaller *student* model, with the goal of training the student model to achieve a similar performance as the teacher model. The idea behind knowledge distillation is that the more complex teacher model has learned a rich representation of the data, which can be transferred to the student model to improve its performance. This is achieved by calculating the Kullback-Leibler divergence (KL) between the teacher outputs  $q^T$  and the student outputs  $q^S$  softened with a temperature factor  $\tau$  in the following way:

$$\mathcal{L}_{KD}(q^T, q^S) = \tau^2 \sum_{i \in I} \sum_{c \in C} \text{softmax}\left(\frac{q_{i,c}^T}{\tau}\right) \log\left(\text{softmax}\left(\frac{q_{i,c}^S}{\tau}\right)\right) \quad (3.2)$$

Li et al. [Li18] used knowledge distillation in Learning without Forgetting (LwF) to mitigate catastrophic forgetting by using the model trained on the previous task as a teacher while the student is learning on new training data. This is achieved by using the output of the previous model as soft labels for the new data. Recent continual learning approaches adapted knowledge distillation in combination with rehearsal, where the distillation loss is typically also applied to the samples of the replay buffer [Reb17, Hou19, Wu19a]. Other approaches additionally include a knowledge distillation term for intermediate activations of the model [Dou20, Zho19] or use different metrics to measure the distance of the output, such as cosine similarity [Hou19]. One of the known limitations of knowledge distillation for continual learning arises

with larger domain shifts between tasks, because the outputs of the teacher will deteriorate the performance of the student [Alj17].

### 3.1.3 Parameter Isolation Methods

Parameter isolation methods mitigate or completely eliminate forgetting by dedicating a subset of a model’s parameters to each task increment, e.g. by masking a set of parameters for each task or by growing new branches for new tasks. However, the main downside of these methods is that simultaneous evaluation for all learned tasks is not possible, as these models require specific paths or parts of the network to be activated for each individual task, which makes these methods impractical for task-agnostic settings. Dynamic architectures are used when there is no constraint on the size of the architecture, so that the model can be expanded for each new task. Early methods relied on creating duplicates of the model for each new task to eliminate forgetting [Alj17] or extending each layer while freezing old weights. However, the number of parameters quickly increases with the number of tasks. Therefore, Ebrahimi et al. [Ebr20] combine their proposed dynamic architecture with replay and disentangle shared and task-specific features with an adversarial loss. Still, these methods often require a task identifier at test time to select the right subset of parameters. DER [Yan21] and Simple-DER [Li21] remove this need by training a single classifier on the concatenation of all embeddings produced by the different parameter subsets. However, when applied to a large number of tasks, these strategies induce a significant memory overhead and necessitate complex pruning as post-processing. DyTox [Dou22] reduces the memory overhead by sharing the encoder and decoder among all tasks and dynamically expanding special tokens to specialize the decoder on a task distribution. This allows for strict control of the parameters expansion, resulting in negligible memory and computation overhead.

Contrary to dynamic architectures, fixed architecture methods isolate a subset of parameters for each task. This can be achieved by masking parameters using iterative pruning [Mal18], by learning selective routing through the network for each task [Fer17] or by storing the parameters for each task

in superposition to each other in the same model [Che19]. To enable the different subsets of parameters, these methods usually require a task identifier at inference. Wortsman et al. [Wor20] circumvent this limitation and infer the task identifier by selecting the subnetwork with the lowest entropy.

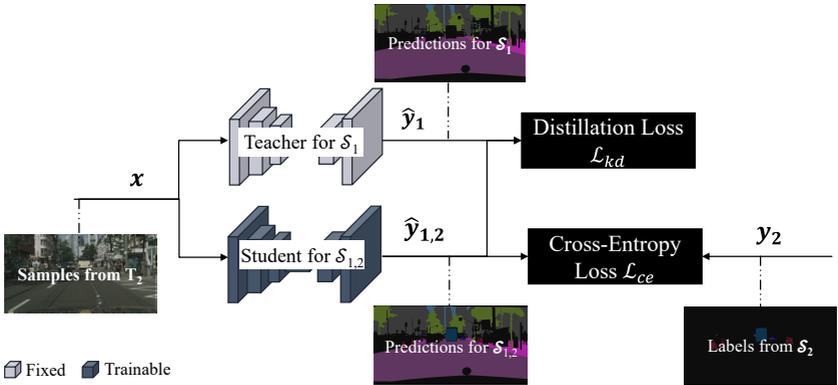
The main downside of parameter isolation methods in continual learning is that they require allocating separate sets of parameters for each task, which can lead to an exponential increase in memory requirements as the number of tasks grows. This can become impractical or even infeasible for large-scale, real-world applications such as automated driving. Additionally, parameter isolation methods do not fully exploit the knowledge gained from previous tasks, which can lead to suboptimal performance on future tasks.

## 3.2 Continual Semantic Segmentation

While continual learning has already been extensively studied for years in object classification, continual semantic segmentation has recently attracted attention, mostly in the topics of class-incremental learning and continual unsupervised domain adaptation. The following section will cover the latest advancements in class-incremental semantic segmentation (CiSS) and domain-incremental semantic segmentation (DiSS).

### 3.2.1 Class-Incremental Semantic Segmentation (CiSS)

In class-incremental semantic segmentation, each new task requires learning a novel set of classes, while previously learned classes remain unlabeled in subsequent tasks. As a result, the model is challenged to learn to distinguish between old and new classes, even though they are never labeled within the same image. State of the art approaches typically address this challenge by implementing a knowledge distillation-based loss [Hin15], in which a model trained on old data generates soft labels for new training data to reduce forgetting [Li18]. Tasar et al. [Tas19] were the first to adapt this approach to class-incremental semantic segmentation, as displayed in Fig. 3.2.



**Figure 3.2:** Schematic Overview of Learning without Forgetting. The teacher model provides soft labels for classes learned in the first task, as in the second task only ground-truth labels for new classes are provided. So the student model learns to distinguish between new and old classes by combining knowledge distillation for old classes and cross-entropy loss for new classes.

Follow-up work by Michieli et al. [Mic19] and Klingner et al. [Kli20] improved this approach by only producing soft labels for the unlabeled part of the new training samples and therefore stopping interference between old and new classes. More recent work explicitly addresses the semantic background shift that occurs between different class increments, for example by de-biasing the initialization of the classifier for the background class when new classes are added and normalizing the objective function to account for the background shift [Cer20]. Douillard et al. [Dou21a] were the first to incorporate a confidence-based pseudo-labeling strategy to integrate hard labels into the training. Finally, Michieli et al. [Mic21] focused on learning more discriminative features by enforcing latent consistency for old classes using prototype matching, while at the same time using a contrastive loss to cluster the features according to semantics.

The effectiveness of these knowledge distillation-based approaches has been demonstrated on their respective benchmarks, in which classes of previous tasks reappear in the background of images of future tasks. However, it should be noted that if this cannot be ensured, these methods are still susceptible

to catastrophic forgetting [Kal21]. To overcome this limitation, several approaches combine replay of old classes with the knowledge distillation-based loss [Mar21, Dou21b]. In these works, data are either replayed as saved full image-label pairs [Mar21, Kal21], by segments of each class [Dou21b] or using unlabeled auxiliary data [Cer22, Yu22]. The most recent approach utilizes the reformulation of semantic segmentation as a mask-classification problem and propose the Continual MaskFormer [Cer22], which uses the MaskFormer [Che21b] architecture with an adaptive distillation loss and a mask-based pseudo-labeling strategy. Contrary to approaches in class-incremental classification, CiSS approaches are mainly based on knowledge distillation. Finally, recent work explores unsupervised class-incremental learning, in which a model learns new unknown classes without annotations by discovering unknown objects in images and clustering them [Uhl22].

### **3.2.2 Domain-Incremental Semantic Segmentation (DiSS)**

The literature on domain-incremental learning for semantic segmentation is limited, as research is mostly focused on adapting to new domains in an unsupervised manner. Recent work in supervised domain-incremental learning by Garg et al. [Gar22] proposes an architecture that learns domain-specific paths in the network for the individual domains to enable the model to learn domain-specific features of each domain, while the majority of the parameters are shared between the domains. Mirza et al. [Mir22a] address the issues of biased population statistics in the batch normalization layers by re-estimating the population statistics for each individual domain. This enables them to use domain-specific population statistics during inference, which prevents the bias toward the most recently learned domain.

However, most of the current research is focused on continual domain adaptation (CDA), in which a segmentation model is first trained on a labeled source domain and then sequentially adapted to a series of unlabeled target domains. During the adaptation process on a new domain, the model can only access the source domain dataset and the most recent target domain dataset [Mar22].

CDA approaches typically preserve information about the style of each domain, e.g. by capturing the style of the target domains using GANs [Wu19b, Mar22] or by storing low-frequency components of each domain [Ter21]. Using this style information, the labeled source images can then be transferred into the styles of the different target domains during training without access to the previous target domain images. Kim et al. [Kim20] propose to save domain-specific information in a small-capacity sub-network during training so that these sub-networks can be used during inference for each target domain. While CDA is a more complex challenge than domain-incremental learning, the challenges of avoiding catastrophic forgetting are the same. A more general overview of domain adaptation for semantic segmentation is provided by Schwonberg et al. [Sch23b].

### 3.3 Effects of Catastrophic Forgetting

The performance of a continual learning method is mostly measured by the accuracy of the model on the individual test sets in a task sequence. Díaz-Rodríguez et al. [Día18] propose three metrics, namely *average accuracy*, which measures the average accuracy on the test set on all tasks, *backward transfer*, which measures the influence that learning a task has on the performance of previous tasks and *forward transfer*, which measures the influence that learning a task has on the performance on future tasks. While these metrics are valuable to evaluate and compare various continual learning algorithms on their respective benchmarks, they cannot provide deeper insights into the specific causes and effects of catastrophic forgetting or which parts of the model are most affected.

Therefore, recent work has begun studying the effects of catastrophic forgetting. Murata et al. [Mur20] measure forgetting for a specific layer  $l$  by measuring *Partial Retrain Accuracy*, which is the accuracy that can be gained after freezing layer  $l$  while the rest of the model is re-trained on all data. Using this method, they show a non-negligible amount of forgetting is already happening at early layers.

Similarly, Davari et al. [Dav22] study representational forgetting using linear probing, which compares the accuracy of an ideal linear classifier before and after introducing a new task. Their results support the view that the observed test accuracy of continual learning algorithms only offers limited insights into the model and indicate that representational forgetting is not as severe as accuracy metrics suggest. Therefore, they advise that it is necessary to evaluate representational changes within the network to avoid misinterpretation of results. Nguyen et al. [Ngu19] used Task2Vec [Ach19] to investigate how properties of the task sequence, such as the similarity and complexity of individual tasks, affect catastrophic forgetting. Surprisingly, they found no correlation between the error rates and the similarity of the tasks in a sequence and a positive correlation between the error rate and task complexity [Ngu19]. Follow-up work for class-incremental learning demonstrated that forgetting is not evenly distributed throughout the model but instead concentrated at later layers, which change significantly, specifically for tasks with intermediate similarity [Ram21].

Other works investigate the loss landscape of the model by using linear mode connectivity [Fra20] to show that different optima obtained by gradient-based optimization are connected by simple paths of non-increasing loss [Mir21] and how different training regimes such as dropout, learning rate decay and batch size widen the local minima to reduce forgetting on subsequent tasks. Recently, De Lange et al. [De 22] identified that common continual learning models suffer from substantial forgetting upon starting to learn new tasks, but that this forgetting is only temporary and followed by a phase of performance recovery. They refer to this phenomenon as *stability gap* and propose continual evaluation for models to measure the worst-case performance of a model.

## 4 Continual Learning for Class- and Domain-Incremental Semantic Segmentation

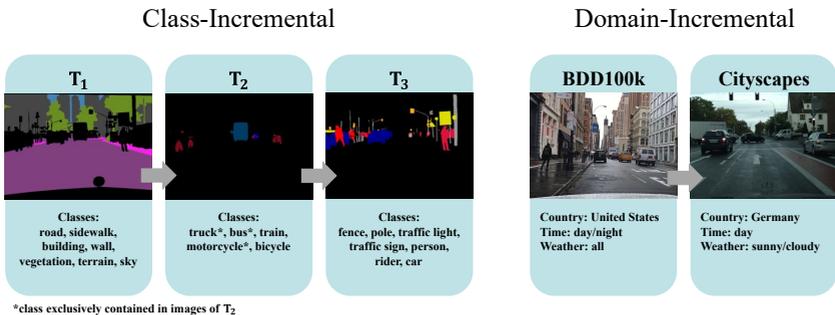
Continual learning research has been mainly focused on developing algorithms for class- and domain-incremental classification tasks [Hsu18, Mas20, De 19], with the underlying assumption that approaches are task-agnostic [De 19], so that results are transferable to other machine learning tasks. However, this assumption may not be entirely valid for tasks like semantic segmentation, which exhibits several fundamental distinctions from classification. Unlike classification tasks, where each image contains only one object of interest, semantic segmentation requires locating and differentiating between different classes in the same image. Especially in class-incremental settings, old classes will typically appear in the background of future training samples, which can lead to additional confusion.

Furthermore, the benchmarks on which continual learning approaches are usually evaluated, such as Split-MNIST [Hsu18], Cifar-100 [Kri09] and ImageNet [Den09], have very specific and controlled domain and class increments. This makes them less representative of the challenges faced in automated driving, where the domain and class increments are more varied and unpredictable. For instance, in automated driving, domain increments could correspond to different weather or lighting conditions, camera perspectives, new higher resolution cameras, or even changes in the road infrastructure. Meanwhile, class increments correspond to new objects, such as e-scooters, different types of road signs or vehicles. To address this gap, this chapter introduces a domain- and class-incremental benchmark for semantic segmentation for scene perception in automated driving based on Cityscapes [Cor16]

and BDD100k [Yu20]. This benchmark is then utilized to evaluate the performance of several continual learning approaches for continual semantic segmentation. Finally, the obtained results are compared to recent continual classification surveys of Hsu et al. [Hsu18] and Masana et al. [Mas20] to gain insights into challenges that might arise specifically for semantic segmentation under more realistic conditions. This chapter is mainly based on the author’s publications [Kal21] and [Kal22b].

## 4.1 Incremental Semantic Segmentation

As described in Sec. 2.4 the goal of semantic segmentation is to assign a class out of a set of pre-defined classes  $\mathcal{C}$  to each pixel in a given image. However, in incremental semantic segmentation, the model  $f$  is not optimized on a single task  $T$  but on a sequence of tasks  $T_k$ , that can introduce new classes or visually distinct instances from the same classes. These differences between tasks can include a shift in the label distribution  $P(Y)$  and/or in the input distribution  $P(X)$ . With this, the base incremental learning scenarios that were presented in Sec. 2.3 can be defined, namely, class-, domain-, and task-incremental learning. For the evaluation in this chapter, the task-incremental setting is disregarded as it requires a task identifier that specifies which task the model is evaluated on during testing, which is not available in many real-world applications. Therefore, this section introduces a class- and a domain-incremental benchmark for semantic segmentation based on realistic driving scenarios from Cityscapes and BDD100k. An overview of these benchmarks is displayed in Fig. 4.1.



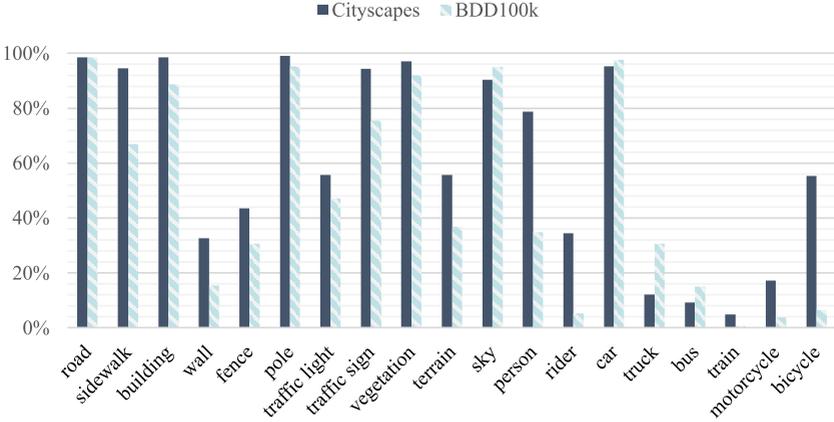
**Figure 4.1:** Class- and domain-incremental learning scenarios for continual semantic segmentation. In the domain incremental setting only the input distribution changes and the classes stay the same. In the class incremental setting the input distribution stays the same, but every new task adds a set of novel classes.

### 4.1.1 Domain-Incremental Segmentation

In the domain-incremental learning scenario, at each increment the conditional distributions of input  $P(X_k|Y_k)$  are changing while the output distribution  $P(Y_k)$  remains the same, so that  $P(X_1|Y_1) \neq P(X_k|Y_2)$  while  $P(Y_1) \approx P(Y_2)$ . In practice, this means that one observes the same classes, but they differ in their visual appearance. Examples of such domain increments would be adapting from synthetically-generated data to camera-recorded data, from one country to another, from day to night, or different weather conditions, while the same selection of classes would be labeled. To meet both requirements, in the domain-incremental setting the models are initially trained on BDD100k and subsequently fine-tuned on Cityscapes.

The first requirement  $P(X_1|Y_1) \neq P(X_k|Y_2)$  is fulfilled as the datasets are recorded in vastly different driving scenarios with varying camera setups. While BDD100k covers diverse driving environments, including day and night drives, driving during different weather conditions, driving on highways and driving in cities in the United States, Cityscapes is collected during the daytime in dry weather conditions across German, Swiss and French cities. Since

the datasets share a common 19-class labeling policy and a similar class distribution,  $P(Y_1) \approx P(Y_2)$ , which can be seen in Fig. 4.2, this setup is purely domain-incremental.



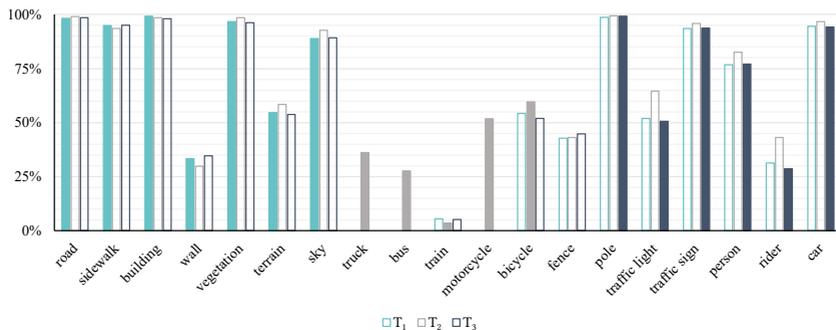
**Figure 4.2:** Class distributions of Cityscapes and BDD100k, measured by the percentage of images containing a specific class (horizontal axis). The plot highlights that Cityscapes and BDD100k have similar distributions of classes.

### 4.1.2 Class-Incremental Segmentation

In the class-incremental setting, each task  $T_k$  extends the previous set of classes  $\mathcal{C}_{k-1}$  by a set of exclusive novel classes  $\mathcal{S}_k$  resulting in the new label set  $\mathcal{C}_k = \mathcal{C}_{k-1} \cup \mathcal{S}_k$ . In this setting, the labels of classes  $\mathcal{C}_{k-1}$  are not part of the training set of the current task  $T_k$ . As each task introduces an exclusive subset of new classes by definition  $\mathcal{C}_1 \neq \mathcal{C}_2$ . In class-incremental classification, this requirement also implies that  $P(X_1) \neq P(X_2)$  because every training sample only contains one class. However, in semantic segmentation the input distribution can remain similar, i.e.,  $P(X_1) \approx P(X_2)$ , as an image contains multiple classes, which include previous and potentially future classes. Therefore, similar to Klingner et al. [Kli20] the Cityscapes dataset is divided into three distinct subsets:  $T_1$ ,  $T_2$ , and  $T_3$ . In each subset, only

an exclusive selection of classes  $\mathcal{S}_k$  is labeled that does not reappear in the labels of the other subsets. However, while the classes of a set are not labeled in subsequent sets, they are likely to reappear in the images of subsequent sets since  $P(X_1) \approx P(X_2)$ . To increase the difficulty of the class-incremental setting, a *disjoint* Cityscapes setup is defined, in which images are sampled so that three of the five classes of  $\mathcal{S}_2$  do not reappear in the training images of the remaining subsets  $T_1$  and  $T_3$ .

This ensures that  $P(X_2) \neq P(X_3)$ . The resulting class distributions of the tasks  $T_1$ ,  $T_2$ , and  $T_3$  are shown in Fig. 4.3. Previous works such as [Mic19, Cer20, Kli20] have proposed similar class-incremental settings for semantic segmentation, but the proposed benchmark introduces an additional challenge by requiring the model to learn classes that appear in  $T_2$  but not in  $T_1$  or  $T_3$ .



**Figure 4.3:** Class distributions of the three defined subsets of the class-incremental disjoint Cityscapes setup. The solid bars in indicate that these classes are labeled in this subset while those represented in empty bars are unlabeled. The classes truck, bus and motorcycle exclusively appear in the images of training set  $T_2$

## 4.2 Experimental Setup

Smaller models with fewer parameters are expected to exhibit more severe forgetting. So in an effort to decrease the need to evaluate continual models on long training sequences, the difficulty of the setting is assessed by comparing

the performance of two models of different sizes. Specifically, this experiment compares the performance of the widely adapted DeepLabV3+ (31M parameters) [Che17b] and the much smaller ERFNet (1.2M parameters) [Rom18]. The results in Tab. 4.1 show that both ERFNet and DeepLabV3+ are capable of learning the task sequences in a non-incremental manner, and as expected, DeepLabV3+ achieves an absolute increase of about 7.9 % in accuracy in the non-incremental setting compared to ERFNet. When learning in the incremental setting, however, it is observed that the drop in accuracy of ERFNet is more severe on the previous task as compared to DeepLabV3+. ERFNet suffers from an absolute drop of 23.9 % mIoU when trained incrementally compared to the non-incremental setting, whereas DeepLabV3+ only drops by an absolute of 16.9 % mIoU. This supports the initial assumption that ERFNet is more susceptible to forgetting due to its smaller size. Hence, the following experiments are conducted with the smaller ERFNet architecture, as this reduces the training time significantly. A more in-depth evaluation of the effects that the architecture choice has on continual learning is given in Ch. 7.

**Table 4.1:** Comparison of DeepLabV3+ (31M parameters) and ERFNet (1.2M parameters) by mean Intersection over Union (mIoU) in the domain-incremental setting. Evaluation is run after training on both datasets either incremental or non-incremental.

Method	Architecture	mIoU <sub>BDD</sub>	mIoU <sub>CS</sub>
Non-Incremental	DeepLabV3+	64.7	77.4
Non-Incremental	ERFNet	57.0	69.4
Fine-Tuning	DeepLabV3+	48.8	77.1
Fine-Tuning	ERFNet	33.1	72.3

### 4.2.1 Baseline Overview

In the following experiments six different continual learning approaches are evaluated that cover the fundamental approaches that recent methods are built on. Specifically, the experiments compare two distillation-based methods (CIL [Kli20], LwF [Tas19, Li18]), three prior-focused approaches (L2 regularization, MAS [Alj18], EWC [Kir15]) and naive replay.

The approach by Tasar et al. [Tas19] adapts the original knowledge distillation-based objective to semantic segmentation, as shown in Eq. (4.1), where the unnormalized score vector  $q^T$  of the teacher is used to replace the one-hot-encoded label  $y$ . The distillation loss is used in combination with the standard cross-entropy shown in Eq. (2.8). The hyperparameter  $\lambda$  is used to balance the losses.

$$\mathcal{L}_{\text{lwf}} = \mathcal{L}_{\text{ce}}(y, \hat{y}) + \lambda \mathcal{L}_{\text{kd}}(q^T, q^S) \quad (4.1)$$

CIL improves LwF by applying the distillation loss only for unlabeled sections within the image to avoid interference with regions for which labels are available.

The prior-focused regularization approaches vary the weights’ plasticity based on the estimated importance  $\Omega_j$  of the weights in previous tasks. The resulting loss is defined by

$$\mathcal{L}_{\text{reg}}(\hat{y}, y, \tilde{\theta}) = \mathcal{L}_{\text{ce}}(y, \hat{y}) + \lambda \sum_j \Omega_j (\theta_j - \tilde{\theta}_j)^2 \quad (4.2)$$

where  $\tilde{\theta}_j$  is the  $j$ -th parameter of the old network and  $\lambda$  is a hyperparameter to balance the regularization term. EWC uses the diagonal of the Fisher information matrix to infer the importance  $\Omega_j$  of a weight. MAS computes the importance  $\Omega_j$  based on how sensitive the predicted output function is to a change in this parameter  $\theta_j$  for a new given sample. In the original MAS implementation, the importance update could also be calculated on arbitrary unlabeled data. However, in order to achieve a fair comparison to the remaining approaches that do not utilize additional data, the training dataset is used instead. Additionally, a naive baseline approach for regularization is evaluated in which all parameters are weighted equally, so that  $\Omega_j = 1 \forall j \in \theta$ , which is termed L2 regularization.

The naive replay approach randomly selects a subset of training samples from past tasks and replays them during training on new data. In these experiments, replay uses a fixed buffer size of either 32 or 64. The memory buffer is evenly shared for each previous task. During training on subsequent tasks,

each training batch is constructed of an equal amount of new data and replay data. Additionally, a combination of CIL and replay is also used in the evaluation, that employs both knowledge distillation and a small replay buffer in order to mitigate forgetting. This combination is denoted as CIL + Replay.

Finally, an upper bound and lower bound of performance are inferred by comparing the results of the continual learning methods to those obtained by training the models in a non-incremental manner and by naively adapting the model to new data. Two methods are compared for naive adaptation: the fine-tuning (FT) approach and feature extraction (FE). While fine-tuning optimizes the parameters of the encoder and decoder of the model, FE freezes the model’s encoder and only optimizes the decoder.

## 4.2.2 Evaluation

All of the following experiments are initialized with the ERFNet [Rom18] model that is pretrained on ImageNet. The model is optimized for 250 epochs with a batch size of 8 using stochastic gradient descent (SGD) with momentum 0.9 and weight decay of  $3 \times 10^{-4}$ . Moreover, the optimizer uses a learning rate of 0.01 and a polynomial learning rate schedule with a power of 0.9. For the second task, the learning rate is decreased to 0.005 for the domain-incremental setting and to 0.007 for the class-incremental setting. During training, the images are first cropped to a 2:1 ratio, then scaled by a factor between 1.0 and 2.0 and finally cropped to a random region of size  $512 \times 1024$ . The network architecture remains unchanged except for regularization and fine-tuning approaches in the class-incremental setting, for which a new decoder head is added for every task, as proposed by [Kli20].

All results are reported using the mIoU on the respective evaluation sets of Cityscapes and BDD100k, averaged over 5 runs each initialized with different random seeds. In order to evaluate approaches in the class-incremental setting, the mIoU is only computed with regard to the classes of the respective subset, so when evaluating on  $T_1$ , the mIoU does not account for classes of  $\mathcal{S}_2$  or  $\mathcal{S}_3$ . However, after completing training on all three subsets, the  $\text{mIoU}_{CS}$

compares the final performance with respect to all 19 classes on the Cityscapes evaluation set.

## 4.3 Results and Discussion

**Table 4.2:** Comparison of different continual learning methods on ERFNet by mIoU for the class-incremental setting. Evaluation is run on the Cityscapes validation set using the indicated class subsets shown in Fig. 4.2. Colors are normalized per column. The mean and standard deviation over five runs is reported.

Method	after Training on $T_1, T_2$		after Training on $T_1, T_2, T_3$			
	$T_1$	$T_2$	$T_1$	$T_2$	$T_3$	mIoU <sub>CS</sub>
Offline			84.8 ± 0.2	76.3 ± 0.9	83.3 ± 0.1	70.8 ± 0.4
FT	50.7 ± 3.1	70.1 ± 2.8	50.7 ± 2.2	49.5 ± 5.3	80.6 ± 0.2	31.1 ± 1.1
FE	79.1 ± 0.0	56.9 ± 0.5	79.1 ± 0.0	56.9 ± 0.5	69.4 ± 0.3	40.7 ± 0.3
L2	79.3 ± 0.2	57.5 ± 1.9	79.0 ± 0.1	56.9 ± 1.9	69.6 ± 0.2	42.3 ± 0.4
EWC	79.3 ± 0.1	57.2 ± 2.7	79.0 ± 0.1	56.9 ± 2.9	68.6 ± 0.3	42.0 ± 0.8
MAS	72.3 ± 1.2	68.5 ± 2.3	73.1 ± 1.1	66.6 ± 1.4	74.3 ± 0.3	42.2 ± 0.7
LWF	77.2 ± 0.7	74.0 ± 2.7	76.9 ± 0.5	71.2 ± 3.6	79.6 ± 0.4	60.0 ± 1.4
CIL	78.5 ± 0.4	75.3 ± 3.3	78.0 ± 0.6	70.6 ± 1.6	79.7 ± 0.4	63.9 ± 0.4
Replay (32)	67.5 ± 1.4	67.2 ± 2.4	66.8 ± 1.5	45.7 ± 4.0	81.8 ± 0.6	42.0 ± 0.9
Replay (64)	71.6 ± 1.9	69.9 ± 1.0	69.2 ± 1.3	50.1 ± 2.6	81.4 ± 0.7	43.3 ± 1.0
CIL + Replay	76.3 ± 1.2	65.4 ± 1.8	76.1 ± 0.6	50.5 ± 4.5	81.7 ± 0.1	57.7 ± 0.3

### 4.3.1 Results on Class-Incremental Learning

The results in Tab. 4.2 show that although FT is better than FE at adapting to new classes it suffers from severe forgetting of old classes, achieving only mIoU<sub>CS</sub> of 31.1 % compared to the 70.8 % when trained offline. On the other hand, feature extraction (FE) completely retains the original performance on the individual tasks but is inhibited in learning the newest classes and struggles to discriminate between classes of different tasks.

The prior regularization approaches perform similarly to FE in terms of retaining performance on the previous task. However, since the weights in the encoder are no longer completely fixed, the model is better at adapting to new classes, resulting in a small increase for the latest classes. It is noteworthy that

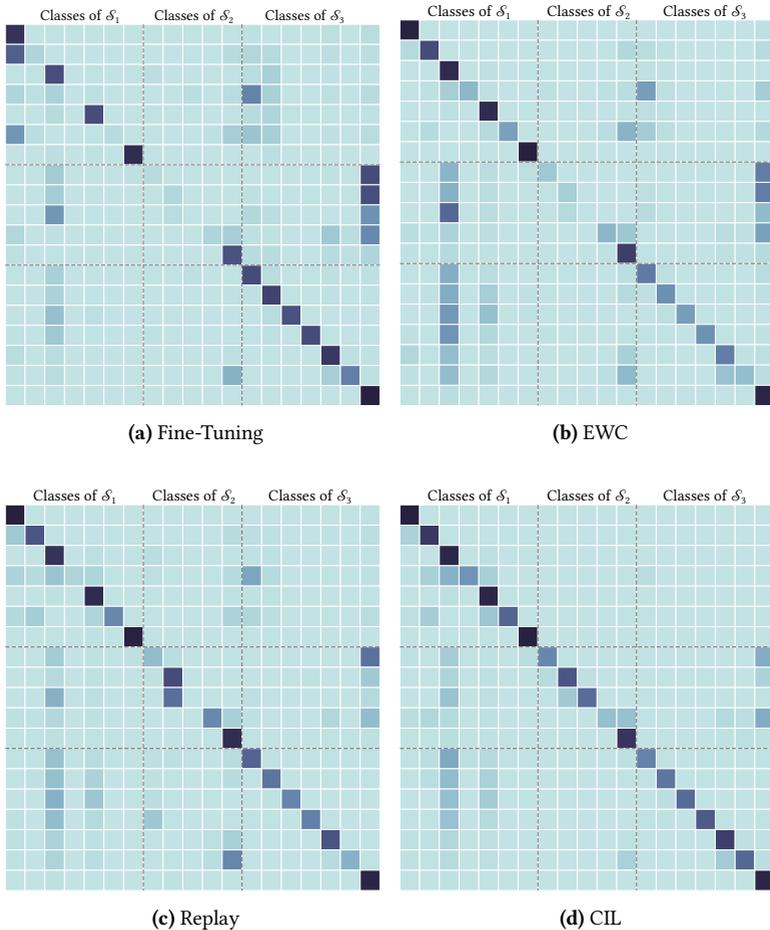
naive L2 regularization performs similarly to the more sophisticated MAS and EWC, which was also observed for classification tasks [Hsu18].

All of the aforementioned approaches score a high mIoU in the range from 50 % to 80 % on the individual subsets  $T_1$ ,  $T_2$  and  $T_3$ , but the overall performance on all classes is only at 40.7 % mIoU, which indicates that these models are not able to distinguish between classes of different tasks. This is likely a combination of a severe bias towards the most recent task and the fact that the models suffer from inter-task confusion as they do not learn discriminating features for the classes of different tasks as they are never observed in the same task. The confusion matrices for FT and EWC in Fig. 4.4 also validate that the classes of  $\mathcal{S}_1$  and  $\mathcal{S}_2$  are mostly confused with the classes of the most recent task. While there is a visible improvement when comparing the confusion matrix from FT to EWC, EWC still classifies the classes  $\mathcal{S}_2$  still mostly as classes from the latest training set  $\mathcal{S}_3$ .

Replay barely outperforms the prior regularization methods when evaluated on all classes, which is unexpected as it is the most effective method in class-incremental classification [Hsu18]. When looking at the confusion matrix in Fig. 4.4 it can be seen that replay is effectively reducing the bias towards the new classes and improves the performance for classes of  $\mathcal{S}_2$ . However, the performance for  $\mathcal{S}_2$  deviates significantly, up to 4 % mIoU, which is likely caused by the random sample selection.

The best performance is achieved by the knowledge distillation-based approaches CIL and LwF, which score the overall highest results, outperforming the next best approach by 18 % mIoU and achieving a  $\text{mIoU}_{CS}$  of 63.9 %. This is attributed to the multi-class nature of semantic segmentation, because old classes can reappear in the background of new training data so that they are effectively rehearsed when using knowledge distillation. However, the confusion matrix shows that once these classes do not reappear, as is the case for  $\mathcal{S}_2$ , the model still struggles to correctly classify those classes. Section 6.1 will investigate this observation in more detail. The combination of CIL and replay achieves worse overall performance than CIL alone, this is likely caused by the fact that the replay samples can contain new classes that are labeled as background, which inhibits learning the new classes.

The overall results show that, contrary to the results for image classification that were reported by Hsu et al. [Hsu18], knowledge distillation is the most effective method for class-incremental segmentation. However, some form of replay will be required for classes that will not reappear in new training data to prevent forgetting of these classes.



**Figure 4.4:** Confusion matrices after training on the Cityscapes Disjoint Sequence.

### 4.3.2 Results on Domain-Incremental Learning

Table 4.3 displays the results for the domain-incremental setting. The results highlight that all continual learning approaches mitigate forgetting in some way compared to fine-tuning, but particularly the regularization-based approaches struggle to adapt to the new task. Overall, FT is best at adapting to the new task, but in turn suffers from the most severe performance drop on the first task. Freezing the encoder drastically reduces forgetting on the first task, leading to an increase of 15 % mIoU for FE compared to FT, but also inhibits adapting to the new task. Similar to class-incremental learning, the prior-regularization approaches (EWC, MAS, L2) achieve a better balance between the adaptability of FT and the stability of FE, as they do not completely prohibit parameter updates of the encoder but simply constrain updates on them. Among the prior-focused methods, EWC is observed to be the best at adapting to new data, while L2 regularization was the most effective at retaining the previous task’s performance.

LwF outperforms the prior-regularization methods in mitigating forgetting on the old task but has similar issues in learning the new tasks, where it is outperformed by FT by 7.8 % mIoU<sub>CS</sub>. This suggests that the teacher-model limits the student’s ability to adapt to new data, which happens if the teacher contradicts the current training labels. This problem does not arise in the class-incremental setting, as the teacher’s predictions are for a disjoint set of classes that occur in a different region of the image; therefore, knowledge distillation is highly effective in the class-incremental setting.

Replay achieves the highest average mIoU of over 59.0 %, which is 4 mIoU below the upper-bound performance of the offline model, while using only 64 images of the original 7000 (0.91 %) images of BDD100k. Small memory sizes are commonly believed to cause overfitting due to the limited diversity of samples. However, the results suggest that severe overfitting is unlikely to occur once the model has been optimized for the entire dataset. A hypothesis is that the combination of new diverse data with the memory buffer keeps the model close to the minima of the previous task, which corroborates the results in Ch. 5 and the findings of Verwimp et al. [Ver21].

However, the standard deviation of the performance on the first task for replay is noticeably increased compared to the other remaining approaches, specifically for the smaller memory buffer size  $n = 32$ . This can be attributed to the fact that replay samples are chosen randomly, which leads to more deviating results in each run. As this is problematic for real applications, Sec. 4.4 discusses further sample strategies for continual semantic segmentation.

**Table 4.3:** Comparison of different continual learning methods on ERFNet by mIoU for the domain-incremental setting. Evaluation is run after training on both datasets either incremental or non-incremental. Colors are normalized per column.

Method	mIoU <sub>BDD</sub>	mIoU <sub>CS</sub>	Average
Offline	57.0 ± 0.4	69.4 ± 0.2	63.2
FT	33.1 ± 0.8	72.3 ± 0.5	52.7
FE	48.3 ± 0.4	58.7 ± 0.1	53.5
L2	46.8 ± 0.7	65.5 ± 0.3	56.2
EWC	46.0 ± 0.7	67.2 ± 0.5	56.6
MAS	44.8 ± 1.2	66.6 ± 0.6	55.7
LWF	48.5 ± 0.4	66.4 ± 0.3	57.5
Replay (32)	45.9 ± 1.8	69.9 ± 0.7	57.9
Replay (64)	47.0 ± 1.1	71.1 ± 0.4	59.0

### 4.3.3 Comparison to Results on Image Recognition

To contextualize the findings so far, they are compared to the findings of Hsu et al. [Hsu18], who extensively evaluated various continual learning techniques for image recognition. Their study identifies two significant similarities with the results of this chapter: firstly, the ineffectiveness of regularization approaches in mitigating forgetting; and secondly, the effectiveness of replay in the domain-incremental setting. However, due to the nature of semantic segmentation, where multiple classes may appear in a single image, the efficacy of other approaches varies in the class-incremental setting. Notably, knowledge distillation appears to be far more effective in CiSS than in classification, as classes are expected to reoccur in the background region of new training images, producing superior results in this evaluation and only slightly outperforming fine-tuning in image recognition. Additionally, the results of this chapter indicate that replay-based methods may face challenges

in distinguishing between classes of different class increments in the context of semantic segmentation, whereas in image recognition, replay is often considered necessary in the class-incremental setting, as reported by [Hsu18, Ven20]. These findings suggest that the background class may play an important role in both causing and mitigating forgetting in CiSS, which Sec. 6.1 will focus on.

## 4.4 Improving Data Selection for Replay

The experiments have shown that replay is a simple yet effective method to combat catastrophic forgetting in domain- and class-incremental semantic segmentation. However, the experiments also demonstrated that the performance of replay varies significantly when samples are selected by random choice, especially when learning in a sequence of class-unbalanced tasks with small memory sizes. Therefore, this section investigates various sample selection strategies for continual learning that aim to outperform random sample selection for the task of semantic segmentation.

In these experiments the following methods are compared to select training samples for the memory buffer:

- Random: samples are chosen randomly
- Entropy (Max/Median): samples with max/median entropy are selected
- Loss: samples with highest/median cross-entropy loss are selected
- Gradient-based sample selection (GSS): samples are selected based on the diversity of the gradients [Alj19]
- Representation-based sample selection (RSS): samples are selected to approximate the learned representations of the training data of previous tasks. Therefore, the activations are extracted by the encoder of the model and projected into a lower dimensional space, using UMAP

[McI18]. The projected activations are grouped into clusters using k-means clustering and the samples that are closest to the cluster centers are selected.

- Class balanced samples: samples that are closest to a uniform distribution of classes are selected
- Class balanced buffer: samples are greedily selected to steadily move the class distribution in the buffer towards a uniform distribution of classes
- Ambivalent classes: samples with the highest number of different classes are selected

#### 4.4.1 Sample Selection for Domain-Incremental Learning

The methods are first evaluated in domain-incremental settings. The results in Tab. 4.4, highlight that: (1) random sample selection is a strong baseline (2) selection by Mean Loss, Mean Entropy and RSS can outperform random sample selection in reducing catastrophic forgetting, but in turn also inhibit learning the new task. However, they still lead to a higher average mIoU across the datasets. From these three methods, only selection by Median Loss and Median Entropy leads to less deviating results on the first task than random sample selection.

Class-balancing methods or GSS perform worse or similar to random selection. Finally, selection by Max Loss leads to a severe degradation in performance, especially on the first task. The reason for the lackluster performance is that Max Loss selected almost exclusively samples of BDD100k that contain erroneous labels, which result in a high loss.

**Table 4.4:** Results in mIoU (%) of sample selection methods for domain-incremental setting with buffer size = 64. Evaluation is run after training on both datasets.

Method	mIoU <sub>BDD</sub>	mIoU <sub>CS</sub>	Average
Random	47.0 ± 1.1	71.1 ± 0.4	59.1
RSS	48.9 ± 1.2	70.2 ± 0.4	59.6
Loss Median	49.9 ± 0.2	69.7 ± 0.3	59.8
Entropy Median	49.3 ± 0.4	69.3 ± 0.4	59.3
GSS	47.2 ± 0.8	70.5 ± 0.5	58.9
Ambivalent Classes	45.6 ± 0.4	69.7 ± 0.4	57.6
Class Balanced Samples	45.5 ± 1.1	70.0 ± 0.3	57.8
Class Balanced Buffer	43.6 ± 0.6	70.0 ± 0.4	56.8
Entropy Max	38.0 ± 3.9	69.6 ± 0.1	53.8
Loss Max	27.8 ± 0.3	68.8 ± 0.0	48.3

#### 4.4.2 Sample Selection for Class-Incremental Learning

In the class-incremental setting replay is used in combination with CIL. The results are displayed in Tab. 4.5. Overall, it is observed that the sample selection strategy plays a more important role in the class-incremental setting, as the class-balancing methods lead to a significant increase of up to 8.2 % mIoU on  $T_2$  while also reducing the deviation of the results. Contrary to the results in domain-incremental learning, class-balancing methods such as class-balanced samples and class-balanced buffer achieve the best performance, whereas selection with RSS or Median Entropy leads to similar results as random selection.

**Table 4.5:** Results in mIoU (%) of sample selection methods for class-incremental setting with buffer size = 64. Evaluation is run after training on both datasets.

Method	after Training on $T_1, T_2, T_3$			
	$T_1$	$T_2$	$T_3$	mIoU <sub>CS</sub>
Random	76.1 ± 0.6	50.5 ± 4.5	81.7 ± 0.1	57.7 ± 0.3
RSS	73.0 ± 0.5	52.7 ± 0.8	81.7 ± 0.2	57.4 ± 0.2
Loss Median	76.3 ± 0.4	47.1 ± 2.7	82.3 ± 0.6	58.4 ± 1.1
Entropy Median	75.7 ± 0.7	48.2 ± 1.1	81.1 ± 0.9	57.3 ± 0.3
GSS	74.1 ± 0.2	49.5 ± 3.2	80.0 ± 0.1	56.6 ± 0.5
Ambivalent Classes	75.0 ± 0.6	49.6 ± 1.1	81.5 ± 0.3	57.4 ± 0.5
Class Balanced Samples	77.0 ± 0.3	58.7 ± 2.1	82.8 ± 0.5	60.5 ± 0.3
Class Balanced Buffer	75.3 ± 0.3	56.8 ± 3.0	81.6 ± 0.1	58.8 ± 0.1
Entropy Max	75.2 ± 0.4	47.8 ± 2.1	81.1 ± 0.2	57.3 ± 0.3
Loss Max	76.3 ± 0.2	52.7 ± 3.9	81.7 ± 0.2	57.4 ± 0.4

### 4.4.3 Different Buffer Sizes

Comparison of the performance for selected sample selection methods with buffer sizes 32, 64 and 128 are displayed in Figs. 4.5 and 4.6. The figure highlights that with decreasing memory size, the sample selection method has a much bigger impact on the final performance. This means that when memory is scarce, sample selection becomes even more critical.

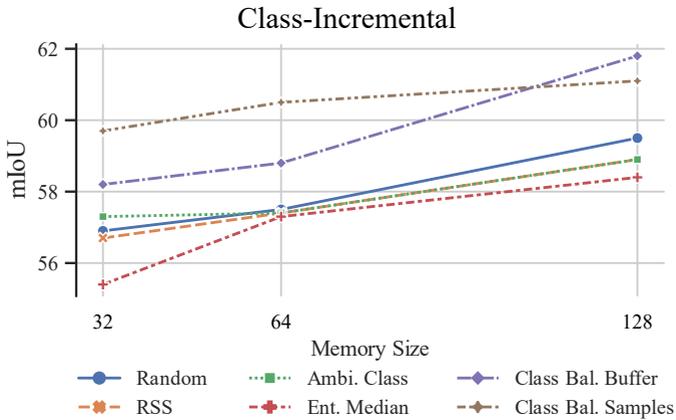


Figure 4.5: Influence of the Memory Size on the performance in mIoU (%) of different sample selection methods in the domain-incremental setting.

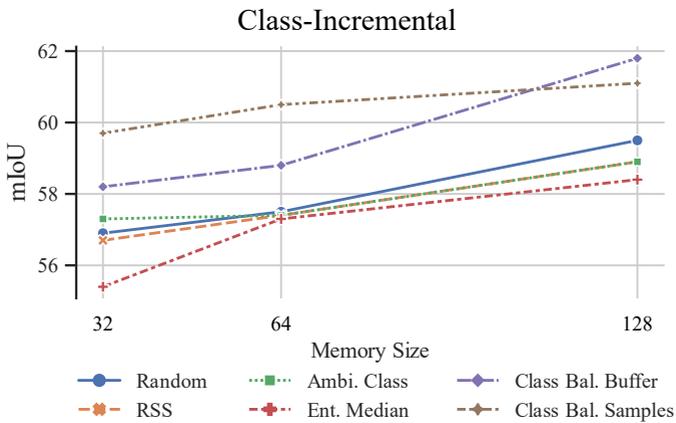


Figure 4.6: Influence of the Memory Size on the performance in mIoU (%) of different sample selection methods in the class-incremental setting.

## 4.5 Conclusion and Research Questions

This chapter compared established methods for continual learning to the task of semantic segmentation and investigated their effectiveness in class- and domain-incremental learning. The results show that the nature of the task of semantic segmentation changes which methods are most effective at mitigating catastrophic forgetting in the class-incremental setting compared to image classification. In particular, knowledge distillation has proven to be most effective at mitigating forgetting in the class-incremental setting. None of the analyzed methods provided satisfactory results for both class- and domain-incremental learning, which indicates that forgetting materializes differently in these settings. Furthermore, the deviating results of the feature extraction approach on the new task suggest that in domain-incremental learning the model is dependent on updating the encoder layers of the network, as freezing these layers has a negative impact on the performance on the new task. While knowledge distillation excels in the class-incremental setting, it is not suited for domain-incremental learning as it struggles to adapt to new data. On the other hand, naive replay is the most effective method to overcome forgetting in domain-incremental learning, only using 32 images (0.46 %) of the previously observed training data, but struggles to distinguish between old and new classes in the class-incremental setting. These observations lead to several research questions, which will be answered in the following chapters.

Section 6.1 will discuss class-incremental segmentation and will answer the following questions:

- 1 Why are regularization methods inefficient in class incremental learning?
- 2 Why is knowledge distillation so much more effective in semantic segmentation than in classification?
- 3 What role does the semantic shift of the background class play in causing and mitigating catastrophic forgetting?

Next, Sec. 6.2 first investigates how domain-incremental learning affects the segmentation model and then compares the differences between the effects in class- and domain-incremental learning:

- 4 How does catastrophic forgetting affect a semantic segmentation model in domain-incremental learning?
- 5 How is forgetting different between class- and domain-incremental forgetting?
- 6 How are batch normalization layers affecting catastrophic forgetting?

Finally, Ch. 7 will investigate the effect of architecture choices to reduce catastrophic forgetting:

- 7 What effect does the neural architecture have on catastrophic forgetting?

As answering these questions requires to measure and locate forgetting at specific parts of the neural network, the next chapter discusses methods that allow deeper insights into the changes within the neural network when trained in a continual manner.

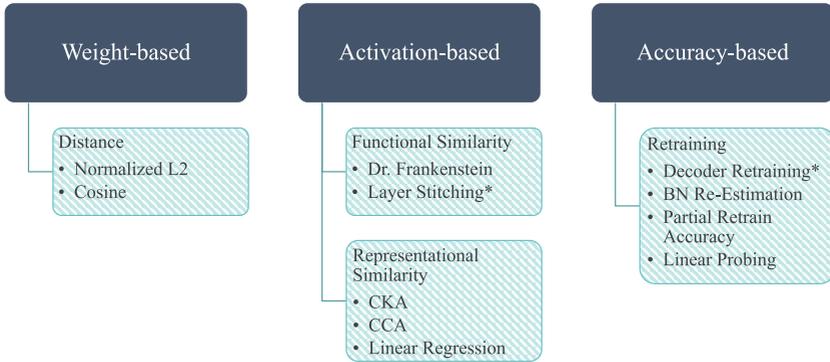
## 5 Methods to Measure the Causes and Effects of Catastrophic Forgetting

The previous chapter demonstrated that catastrophic forgetting might manifest itself differently in semantic segmentation and classification as well as between class- and domain-incremental learning. However, the current means of evaluating continual approaches are limited to evaluation on their respective test sets and cannot provide deeper insights or explanations into the observations made in the previous chapter.

Therefore, this chapter presents a set of tools to measure the causes and effects of catastrophic forgetting in an incrementally trained model for semantic segmentation. These methods allow to measure the representational similarity, weight distance, and inter-task confusion of the model to examine the causes and effects of catastrophic forgetting. In addition, the caveats associated with these techniques are laid out so that incorrect inferences can be avoided. These tools provide the basis for the evaluation of continual semantic segmentation in class-incremental learning and domain-incremental learning in Ch. 6. The presented results in this chapter are mainly based on two of the author's publications [Kal22a, Kal23a].

### 5.1 Overview

As the previously underlying effects of catastrophic forgetting cannot be measured exclusively by the accuracy achieved on the test sets, several methods were proposed to study the causes and effects of catastrophic forgetting. An overview of the methods is displayed in Fig. 5.1.



**Figure 5.1:** Overview of methods that are used to measure the effects of catastrophic forgetting in continual learning. Methods marked with \* were proposed in the author’s publication [Kal22a].

The methods can broadly be divided into three categories: weight-based, activation-based and accuracy-based. Weight-based methods measure the effects of forgetting by computing the distance of models’ weights, either holistically or per layer, to infer which layers are most affected [Ney20]. The downside of these approaches is that they do not take the data distribution into account, as not every weight change implies meaningful changes of the activations with respect to the data distribution.

In comparison, activation-based approaches measure the similarity directly through the activations of specific layers of the network and therefore also take the data distribution into account. Methods like Centered Kernel Alignment (CKA) [Kor19] were already used to measure the activation drift between models of different tasks [Dav22, Ram21]. Activation-based approaches can be further distinguished into approaches that measure representational similarity (e.g. CKA) and approaches that measure functional similarity. Representational similarity measures the change of the activations directly without taking into account the effect that this change has on the prediction of the network, whereas functional similarity methods such as

Dr. Frankenstein [Csi21] gauge the effect that the activation drift has on the output and overall performance of the network on a specific task.

Accuracy-based methods re-estimate or re-train specific parts of the network while keeping the remaining parts of the network fixed to infer how useful the representations of the incrementally trained model are to solve the joint task of the network [Mur20, Dav22]. As there is no prior work on comparing the different tools to measure forgetting in continual learning, the goal of this chapter is to compare the different analysis methods in their ability to measure the effects of catastrophic forgetting in class- and domain-incremental semantic segmentation. In order to understand what inferences can be made when using them and when they potentially lead to wrong conclusions.

## 5.2 Experimental Setup

To allow comparison of these analysis methods in their ability to measure the effects of catastrophic forgetting in semantic segmentation, in these experiments the segmentation architecture is trained in a class- and domain-incremental setting.

In the domain-incremental setting, the model is trained incrementally first on Cityscapes [Cor16] (CS) and then on the ACDC-*Night* [Sak21] subset. ACDC and CS are both large-scale datasets for semantic understanding of urban street scenes for autonomous driving and share a common 19-class labeling policy, so that the increment is purely the change from day images (CS) to night images (ACDC). For the class-incremental setting, the models are trained on the commonly used PascalVOC-15-5 [Mic19] benchmark. The PascalVOC-15-5 split is a two-step incremental learning task that consists of learning 15 classes (1–15) in the first step  $T_0$  and the remaining 5 classes (16–20) in the second step  $T_1$ .

The same ERFNet [Rom18] architecture is trained in these incremental benchmarks using various continual learning methods, namely: fine-tuning (FT), the prior-regularization method EWC [Kir15], Learning without Forgetting (LwF) [Li18, Tas19] and replay. The test results that are displayed in Tab. 5.1,

generally demonstrate the same trends as in Ch. 4: (1) Higher forgetting for fine-tuning in the class-incremental setting; (2) LwF achieves better performance than replay in class-incremental learning; (3) Replay is the most effective method in domain-incremental learning. In the following sections, the models will be thoroughly evaluated using the given tools. In every section a method will be discussed by first giving a general explanation of the method then discussing the results when using the method to measure catastrophic forgetting and finally pointing out the method’s limitations.

**Table 5.1:** Comparison of EWC, replay and fine-tuning in the class- and domain-incremental learning scenarios. Evaluation is run after training on the entire task sequence.

Method	Class-Incremental			Domain-Incremental		
	0-15	15-21	Forgetting	Cityscapes	Night	Forgetting
Fine-Tuning	4.6	22.2	51.3	37.1	41.7	31.3
EWC	33.5	8.1	22.4	39.2	36.5	29.2
Replay	41.0	31.5	14.9	58.2	40.5	10.2
LwF	43.5	25.7	12.4	40.2	39.3	28.2

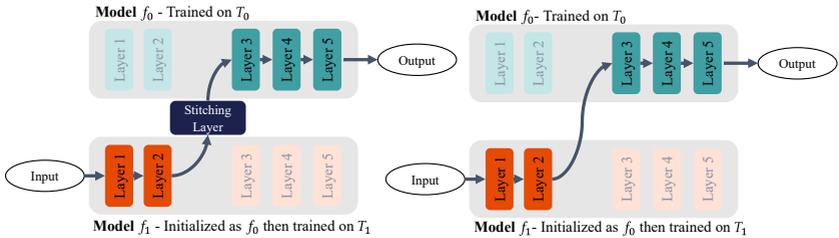
## 5.3 Activation Drift

First, methods to measure the activation drift between two models  $f_0$  and  $f_1$  are evaluated, in which model  $f_0$  is trained on  $T_0$  and  $f_1$  is initialized with the parameters of  $f_0$  and incrementally trained on  $T_1$ . The goal of the methods in this section is to measure the activation drift for every layer  $n$  between the activations  $\mathbf{A}_{0,n}$  and  $\mathbf{A}_{1,n}$  of the models  $f_0$  and  $f_1$ . The current key methods to measure activation drift in neural networks are centered kernel alignment (CKA) [Kor19] and layer matching with Dr. Frankenstein [Kal22a, Csi21]. This section discusses how these methods measure activation drift in continual learning and what the differences between those methods are.

### 5.3.1 Layer Matching with Dr. Frankenstein

The Dr. Frankenstein tool set aims to analyze the similarity of representations in deep neural networks, by matching the activations of two networks

on a given layer by joining them with a  $1 \times 1$  convolutional stitching layer, as displayed in Fig. 5.2. It was first proposed by [Len15] and recently was reevaluated in comparison with other similarity metrics [Csi21, Ban21].



**Figure 5.2:** Comparison of the original Dr. Frankenstein layer matching [Csi21] (left) with the approach without an additional stitching layer (right), named layer stitching.

The goal of the stitching layer is to transform the activations of a specific layer of  $f_0$  to the corresponding activations of model  $f_1$ . The stitching layer is initialized using least-squares matching and optimized using the loss function the network was trained with. In order to measure the similarity of the learned representations, the accuracy of the resulting Frankenstein network is evaluated on the test set and compared to the initial accuracy of the model  $f_0$ . The higher the resulting relative accuracy, the closer the learned representations of the models are to each other.

Contrary to prior work, in which models from completely different initializations are compared, the objective in this context is to measure the activation drift between a model at different points during a continual learning sequence. These models are closely related because  $f_1$  is initialized with the parameters of  $f_0$ . Therefore, the setup shown on the right in Fig. 5.2 without an additional stitching layer is followed, meaning that the activations of the layer  $n$  under examination  $f_1$  are directly propagated to the layer  $n + 1$  in  $f_0$ .

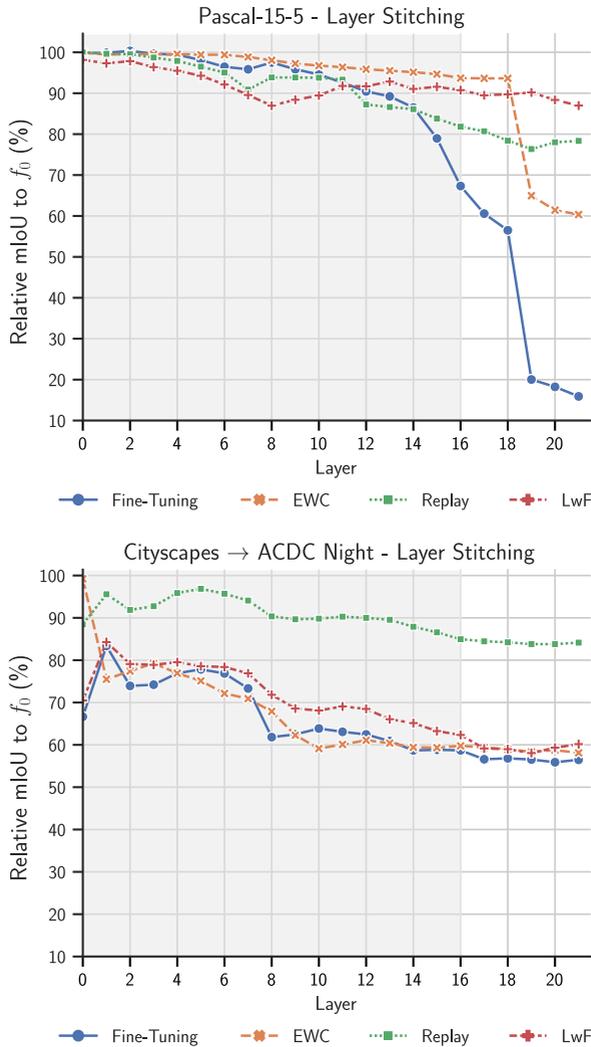
If the accuracy of the stitched network is not adversely affected, this is a clear indication that the internal representations of  $f_1$  were not altered drastically during training on  $T_1$ . This analysis will give insights into how much the activation drift at a specific layer impacts the performance on a previous task. In

the following, “Dr. Frankenstein” refers to when an additional stitching layer is used and “layer stitching” refers to when the additional  $1 \times 1$  convolution layer is omitted.

## ***Results***

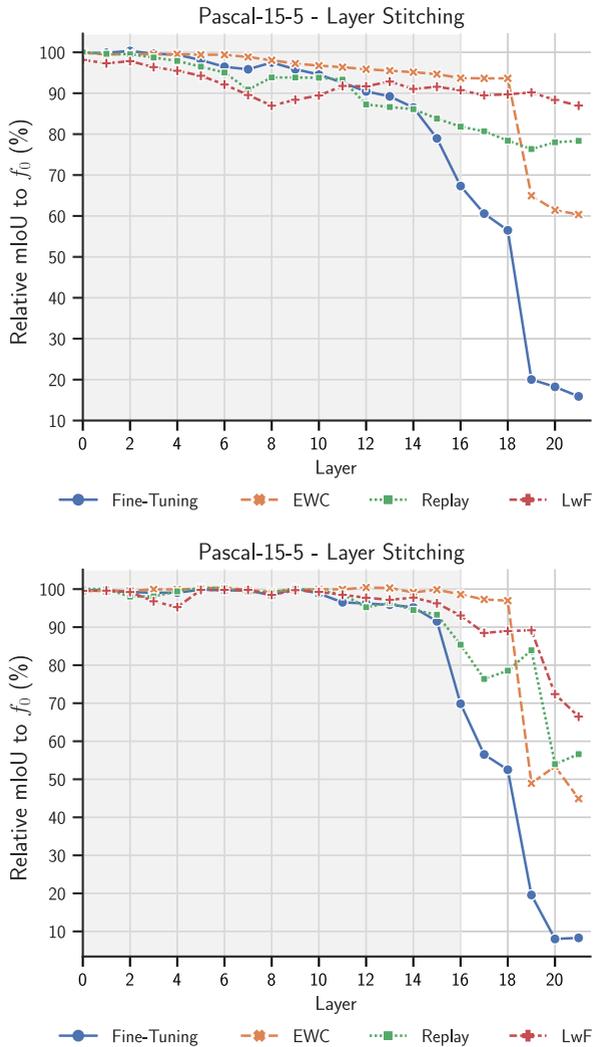
The layer-wise activation drift measured with layer stitching for the incremental learning scenarios is displayed Fig. 5.3. It is apparent that in the class-incremental scenario (Pascal-15-5) the encoder layers up until layer 8 are not at all affected by activation drift, as even after training on a set of new classes in the second task, the model’s representations are completely reusable for the model trained only on the first task. Only later encoder layers and especially the decoder layers show significant representation drift, in which model  $f_0$  is not able to correctly classify old classes using the representations of  $f_1$ . Most notably is the drop in similarity from layer 18 to layer 19, which is observed for fine-tuning and EWC, which will be examined in Sec. 6.1. These results support the conclusions of recent research which found that representation drift in class-incremental learning is concentrated near the classification layer [Dav22, Ram21].

However, in the domain-incremental learning setting, it is apparent that the first layers are primarily affected by activation drift and later layers only change slightly, indicating that the activation drift might be concentrated in the early layers, which Sec. 6.2 will investigate in more depth.



**Figure 5.3:** Activation drift between  $f_1$  to  $f_0$  measured by relative mIoU on the first task of the networks stitched together at specific layers (horizontal axis). The layers of the encoder are layer 0–16 (grey area), the decoder layers are 17–21 (white area). In the class-incremental Pascal-15-5 setting, the activations of the early layers of the encoder stay very stable for all methods, only EWC and fine-tuning have a severe drift in activations in the decoder layers of the network. In the domain-incremental learning setting only the first layers (0–8) are affected by activation drift and layers 9–20 layers only change slightly.

Figure 5.4 compares layer stitching with and without the additional  $1 \times 1$  convolutional layer. The stitching layer is initialized with least-squares matching and optimized using the task loss [Csi21]. With the exception that Dr. Frankenstein generally achieves a higher similarity with the additional stitching layer, both variants show the same trends: high similarity in the encoder; sudden drop of similarity from layer 18 to 19 for EWC and fine-tuning and highest similarity at the final layer for LwF. However, the drop in similarity is more severe when measuring using Dr. Frankenstein compared to layer stitching. This is unexpected as the model without an additional stitching layer achieves a higher performance for layers 19 and 20 and one would expect that the additional stitching layers of Dr. Frankenstein should lead to an increase in similarity as the stitching layers are able to learn additional transformations to transform the activations back to more useful activations for the initial network.



**Figure 5.4:** Activation drift between  $f_1$  to  $f_0$  measured by Layer Stitching (top) and Dr. Frankenstein (bottom) with the additional  $1 \times 1$  convolutional stitching layer.

### Limitations

A limitation of this approach without the additional stitching is that it cannot measure positive backward transfer without the additional stitching layer, in which the model would learn a new, improved representation for old data while learning a new task. This could occur when a feature that was discriminative for  $T_0$  is replaced with a feature that is more useful for discriminating all classes. In that case,  $f_0$  could no longer extract useful information from the stitched representations of  $f_1$ , which would lead to a performance drop, although these representations are still useful for  $f_1$  to classify all classes.

### 5.3.2 Centered Kernel Alignment (CKA)

CKA [Kor19] is a similarity index that measures the similarity between internal representation of neural networks. Given  $|T|$  samples and their corresponding matrices of activations  $\mathbf{A}_n \in \mathbb{R}^{|T| \times p}$  and  $\mathbf{B}_n \in \mathbb{R}^{|T| \times p}$  of  $p$  neurons at a specific layer  $n$  of two neural networks  $f_a$  and  $f_b$  linear CKA is defined as:

$$CKA(\mathbf{A}_n, \mathbf{B}_n) = \frac{\|\mathbf{A}_n^T \mathbf{B}_n\|_F^2}{\|\mathbf{B}_n^T \mathbf{B}_n\|_F \|\mathbf{A}_n^T \mathbf{A}_n\|_F} \quad (5.1)$$

$\|\cdot\|_F$  denotes the Frobenius norm. The higher the score, the more similar are the activations at hand. The value ranges between 0 and 1. Kornblith et al. [Kor19] designed CKA to be invariant to invertible orthogonal transformations and isotropic scaling but not invertible linear transformations. They empirically validate that with these properties, CKA captures intuitive notions of similarity, such as that networks that were trained from different initializations on the same data should be similar, while networks trained on a different dataset or that are completely untrained should be less related. Recently, linear CKA has been used to compare the intermediate representations of models  $f_0$  and  $f_1$  in continual learning [Dav22, Ram21], in which a high CKA score equates to lower representational forgetting.

Csiszárík et al. [Csi21] investigated the relationship between representational similarity that is measured by CKA and functional similarity measured by Dr. Frankenstein. In this case, functional similarity denotes that the representation leads to a similar output of the model, whereas representational similarity directly measures the distance between representations. They demonstrate that a network can retain high functional similarity using layer stitching while simultaneously decreasing the similarity index measured by CKA. This can be explained by the fact that CKA can be affected by changes to features that are not relevant for the networks' predictions. In other words, they can change the representations of a layer while the output of the entire network is not affected notably.

## **Results**

The similarity scores between the representations of  $f_0$  and  $f_1$  measured by CKA are displayed in Fig. 5.5. In the class-incremental setting, it can be observed that the CKA score stays close to 1.0 up until layer 10, at which a decrease in similarity can be observed, specifically for fine-tuning and replay. As one would expect in the class-incremental setting, the similarity scores drop severely in the decoder layers. However, EWC retains much higher representational similarity than replay or LwF, which at first sight contradicts the results in Tab. 5.1, where EWC shows significantly more forgetting than replay or LwF. When combining these findings with the observations from the layer stitching plot in Fig. 5.3 and the results from Tab. 5.1, it can be deduced that the small representational changes, as measured by CKA, lead to significant functional changes, measured by layer stitching, that in turn lead to catastrophic forgetting.

The CKA plot in Fig. 5.5 at the bottom for the Cityscapes to Night setting also illustrates that the similarity of the representations is mostly affected in the early layers and only decreases very slowly throughout the network. Similar to layer stitching, replay shows the fewest changes in the activations.

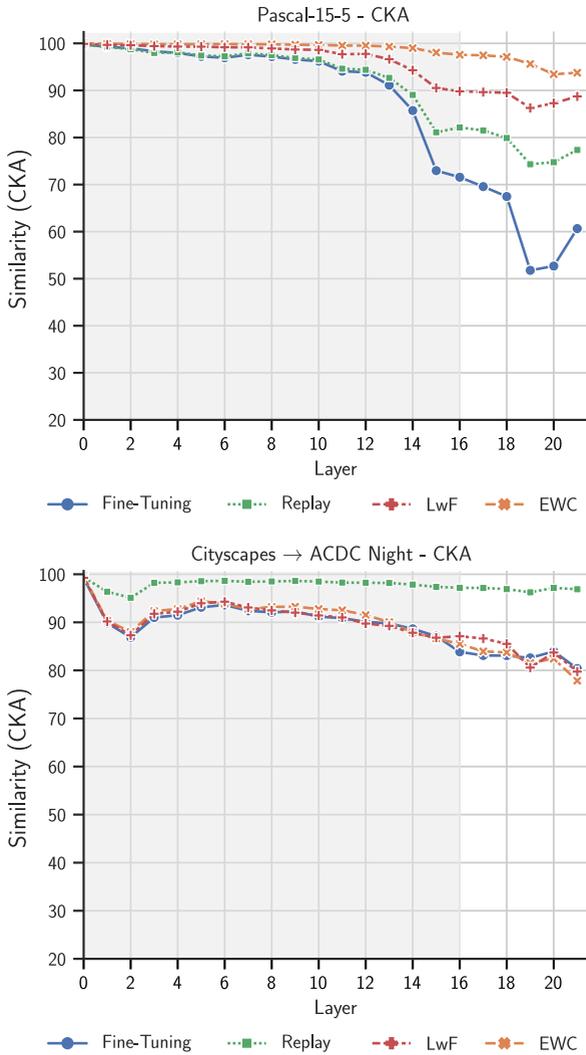


Figure 5.5: Activation drift between  $f_1$  to  $f_0$  measured by CKA for Pascal-15-5 (top) and Cityscapes → Night [Kal22a] (bottom) at specific layers (horizontal axis).

### ***Limitations***

Similar to layer stitching, CKA is also not able to measure positive backward transfer. Furthermore, CKA should be understood as a pure distance measure between feature vectors that does not differentiate between relevant and irrelevant features for the given task. Hence, meaningless changes to the activations that do not have an impact on the predictions can lead to dissimilarity, as observed in the prior experiment. This effect was also observed by Bansal et al. [Ban21]. Therefore, the similarity scores of layer stitching are more meaningful to measure the effects of catastrophic forgetting, as they are not affected by unrelated feature changes [Csi21, Ban21], but still, CKA can be considered a useful complement to layer stitching.

## **5.4 Re-Training and Re-Estimation**

Re-training and re-estimation methods try to freeze specific layers of the network and re-train the remaining layers on the joint data distribution to show how useful the features of the frozen layers are to solving the joint task. Decoder re-training and linear probing freeze the backbone of the model and re-train the classification layer or decoder of the network to estimate how discriminative the features of the backbones are. Partial re-training accuracy on the other hand, measures forgetting for single layers of the network while the remaining are trained from scratch [Mur20]. Finally, batch normalization re-estimation is used to measure the contribution of the changing batch normalization population statistics to catastrophic forgetting.

### **5.4.1 Batch Normalization Re-Estimation**

Major contributors to the activation drift of a model trained incrementally are the changing population mean and variance of batch normalization layers, which are collected during training to achieve a deterministic behavior for

inference [Iof15]. While this works for i.i.d.<sup>1</sup> data, in the non-i.i.d. incremental learning setting, the batch normalization estimates of the population mean and variance are heavily biased towards the most recent task, leading to a significant drop in accuracy on old tasks [Lom20]. A straightforward method to measure the impact of changing BN statistics is to re-estimate them on the joint dataset. This can be achieved by simply doing a forward pass over the entire joint dataset, without the backward pass. After re-estimation of the BN statistics, the model is evaluated on the test set on the first task and the change in mIoU compared to the initial performance on the first task is reported as  $\Delta\text{mIoU}_{BN}$ . This gives a quantifiable measure of how much the changing BN statistics impact catastrophic forgetting. A higher value of  $\Delta\text{mIoU}_{BN}$  means that the network was more affected by the changing population statistics.

## **Results**

Table 5.2 shows the respective re-estimation results for the domain- and class-incremental experiments. By comparing the  $\Delta\text{mIoU}_{BN}$ , it appears that the changing BN statistics have a much more significant impact on domain-incremental learning. Furthermore, in the domain-incremental setting, replay alleviates the change in BN statistics completely, as re-estimation even slightly decreases the mIoU. Therefore, it is concluded that changing BN statistics are a significant contributor to forgetting in the domain-incremental setting and the severity of that effect can be measured with BN re-estimation.

---

<sup>1</sup> independent and identically distributed

**Table 5.2:** Performance in mIoU (%) of the adapted model  $f_1$  after re-estimating the population statistics of all batch normalization layers. By measuring and comparing the increase after re-estimating BN statistics ( $\Delta\text{mIoU}_{BN}$ ), it is observed that in class-incremental learning re-estimating BN statistics leads to a less significant increase compared to the domain-incremental setting.

Method	Class-incremental		Domain-Incremental	
	mIoU <sub>BN</sub>	$\Delta\text{mIoU}_{BN}$	mIoU <sub>BN</sub>	$\Delta\text{mIoU}_{BN}$
Fine-Tuning	4.6	0.0	47.2	10.1
EWC	35.6	2.1	54.1	14.9
Replay	41.2	0.3	56.8	-1.4
LwF	42.1	-1.4	47.0	6.8

### Limitations

BN re-estimation can only give a measure of which batch normalization layers are affected by the changing BN population statistics, but provides no insights into the direct effects of the change. However, it can be vital to understand how a continual learning algorithm is affecting the BN statistics, e.g., how replay stabilizes population statistics using the replay buffer. Finally, it should be noted that this method is not applicable to the recent vision transformer architectures [Dos21], as they use layer normalization [Ba16] instead of batch normalization.

#### 5.4.2 Partial Re-Training Accuracy (PRA)

Murata et al. [Mur20] measure representational forgetting of a given layer  $l_{k,i}$  with partial re-train accuracy (PRA), which is the improvement in accuracy that can be achieved, when re-initializing and re-training the model on all data while the parameters of the given layer  $l_{k,i}$  are frozen. After that, they re-order the sequence in which the tasks are learned to prevent the effect the task order has on the learned representations. Using this strategy, they demonstrate that a significant amount of forgetting is already occurring at shallow layers.

### ***Limitations***

The validity of this method is questionable because the majority of the network is re-trained on the joint data of the model, so the activation drift of intermediate layers can potentially be rectified by following layers as they are trained on the joint task. E.g., when freezing only the very first block of a network in the domain-incremental setting, the remaining layers will amend the activation drift of the first layer, although the very first layers are known to be causes of severe forgetting in this setting. So while the aforementioned approaches, which directly compare the activations, are not able to distinguish whether the model has learned a new representation for old data or if the previous representation has been overwritten, PRA can falsely lead to the conclusion that a new representation has been learned due to re-training.

### **5.4.3 Decoder Re-Train Accuracy and Linear Probing**

Decoder re-training, which the author proposed in [Kal22a] and linear probing [Dav22] aim to measure representational forgetting by calculating the difference in accuracy an optimal classifier layer achieves on an old task before and after introducing a new task. Since the methods are similar, except that decoder re-training is intended for semantic segmentation and linear probing for classification, this section only considers decoder re-training. To measure the decoder re-training accuracy, the encoder of the model is frozen while the decoder is re-trained on all classes with the same training configuration and subsequently evaluated on all tasks. The performance of the re-trained model is denoted as  $mIoU_R$  and the gain as  $\Delta mIoU_R$ . The  $mIoU_R$  shows how useful the learned representations are to discriminate between classes of different tasks. Linear probing and decoder re-training have been used to show that continual learning methods that seem ineffective in the class-incremental setting, such as EWC, are in fact able to stabilize internal representations and that only a few final layers are the main contributors to deteriorating performance on the old task [Kal22a, Dav22].

### ***Limitations***

Decoder re-train accuracy and linear probing are aimed at differentiating between representational forgetting in the encoder and forgetting in the decoder. They indicate how discriminative the features of the backbone are in distinguishing all observed classes and thereby give a quantitative measure on the effect of inter-task confusion. However, they cannot give further insights into which layers are affected. Furthermore, it is not as useful in the domain-incremental setting because forgetting is mainly affecting the early layers.

## **5.5 Weight Drift**

Instead of measuring activation drift, it is also possible to measure the changes of the model from  $f_0$  to  $f_1$  simply by calculating the  $\ell_2$ -distance of the model’s normalized parameters from  $\theta_0/\|\theta_0\|_2$  to  $\theta_1/\|\theta_1\|_2$ , as it was done by Neyshabur et al. [Ney20] to measure the distance between initializations of pre-trained and randomly initialized models. Furthermore, to see how individual layers are affected by weight drift, the distance between specific convolutional layers or batch normalization layers is also measured.

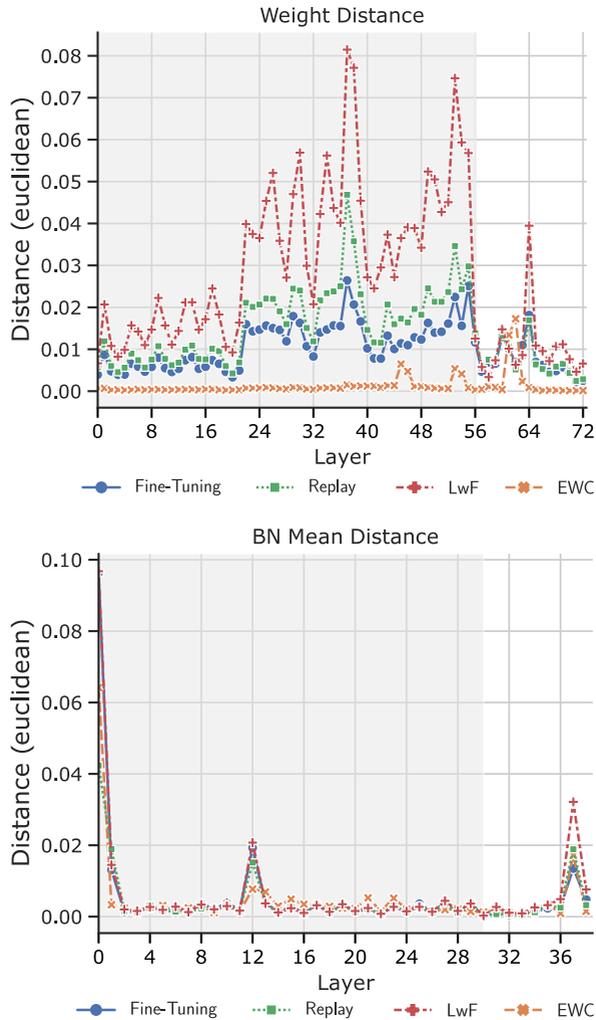
### ***Results***

Table 5.3 displays the  $\ell_2$ -distance for all the models’ parameters for the class- and domain-incremental setting. It is noticeable that the models trained with EWC stay the closest to  $\theta_0$ , which is intuitive as EWC explicitly constrains updates on existing parameters. Although replay and LwF are least affected by catastrophic forgetting, they have the largest  $\ell_2$ -distance between the model’s weights in the class-incremental setting. This clearly indicates that  $\ell_2$ -distance of the weights does not have a meaningful direct connection to catastrophic forgetting.

**Table 5.3:** Weight distance calculated as  $\ell_2$ -distance of the models parameters  $\theta_0$  to  $\theta_1$ . The distance between the models' parameters is lowest for EWC as it explicitly constraints updates on existing parameters. While replay performs the best at mitigating catastrophic forgetting, the weight distance is the biggest for replay. This indicates that weight distance does not always correlate with catastrophic forgetting.

<b>Method</b>	<b>Class-Incremental</b> $\ell_2\left(\frac{\theta_1}{\ \theta_1\ _2} - \frac{\theta_0}{\ \theta_0\ _2}\right)$	<b>Domain-Incremental</b> $\ell_2\left(\frac{\theta_1}{\ \theta_1\ _2} - \frac{\theta_0}{\ \theta_0\ _2}\right)$
Fine-Tuning	0.11	0.2
EWC [Kir15]	0.03	0.16
Replay	0.14	0.25
LwF	0.31	0.23

Additionally, Fig. 5.6 compares the layer-wise distances of the convolutional layers. Interestingly, the model trained with EWC is affected by very minor changes to the models' weights up until later layers in the decoder of the network, which coincides with the layers, in which a significant drop in similarity was observed when using layer stitching. Moreover, the different methods follow a similar pattern of spikes in the Fig. 5.6, though with a different scale. In the case of LwF, that could be potentially explained by the use of knowledge distillation in the loss function, as the temperature value  $\tau$  in Eq. (3.2) changes the scale of the output of the network and therefore also the weights.



**Figure 5.6:** The distance of  $f_1$  and  $f_0$  measured by  $\ell_2$ -distance of the weights of the convolutional layer in the class-incremental setting (top) and the running mean of the batch normalization layers in the domain-incremental setting.

Finally, Fig. 5.6 also displays the  $\ell_2$ -distance of the population mean of the batch normalization layers in Fig. 5.6 for the domain-incremental setting. It is apparent that the very first batch normalization layer undergoes the most drastic changes. After this initial peak, only smaller peaks at very specific layers are observed that coincide with the downsampler block in the encoder and an upsampler block in the encoder.

### ***Limitations***

The major difference between measuring weight drift instead of activation drift is that weight drift does not take the training data into account. However, the results clearly highlight that the distance of the parameters of the model is not indicative of the performance drop on the previous task. Therefore, it is concluded that it can be used to interpret how the weights have changed, but it should not be understood as a direct measure for catastrophic forgetting.

## **5.6 Conclusion**

This chapter evaluated and discussed tools to assess the effects of catastrophic forgetting. A series of class- and domain-incremental experiments showcased the strengths and weaknesses of these tools. It was found that these approaches work best in combination since they complement each other and capture different effects. For example, measuring activation drift with CKA or layer stitching is helpful to locate forgetting, but BN re-estimation and decoder re-training are required to identify the causes. Furthermore, the results illustrate that evaluating weight distances does not correlate with the drop in performance of previous tasks and should not be interpreted as a measure of catastrophic forgetting. Finally, measures of activation drift such as layer stitching and CKA are useful in both domain- and class-incremental settings, whereas BN re-estimation is more insightful in domain-incremental learning and Decoder re-training in class-incremental learning. The summary of the results in Tab. 5.4 again highlights that catastrophic forgetting is nuanced as

many effects contribute to forgetting in a different manner, so that there is no single measure that can show the full picture.

**Table 5.4:** Comparison of methods to measure catastrophic forgetting. As CKA and layer matching measure similarity of activations for every layer, only the minimum values are reported. Values in bold indicate the method that supposedly is least affected by forgetting according to the measure used.

Method	Class-Incremental					Domain-Incremental				
	Forgetting	CKA	Layer Stitching	Weight Distance	$\Delta\text{mIoU}_{BN}$	Forgetting	CKA	Layer Stitching	Weight Distance	$\Delta\text{mIoU}_{BN}$
Fine-Tuning	51.3	<b>51.7</b>	15.9	0.11	0.0	31.3	80.4	55.9	0.20	10.1
EWC	22.4	<b>93.4</b>	60.4	<b>0.03</b>	2.1	29.2	<b>77.9</b>	58.1	<b>0.17</b>	14.9
Replay	14.9	74.3	76.3	0.15	0.3	10.2	<b>96.2</b>	<b>83.8</b>	0.25	<b>-1.4</b>
LwF	12.4	86.3	<b>87.0</b>	0.32	<b>-1.4</b>	28.2	79.8	58.1	0.23	6.8

However, this analysis demonstrates that when these methods are used individually, layer stitching provides the most meaningful and easily interpretable results, as it: (1) takes the data distribution into account, (2) is robust towards irrelevant activation changes compared to CKA and (3) gives an easily interpretable measure for every layer. Therefore, layer stitching will be utilized as a foundation in the following chapter of this thesis to gain a holistic overview of the activation drift within the network and the remaining methods will be applied to understand which specific effect is at play.



## 6 Principles of Catastrophic Forgetting in Continual Semantic Segmentation

The findings presented in Ch. 4 for continual semantic segmentation suggest that forgetting in class- and domain-incremental semantic segmentation may have distinct underlying causes and consequences. Beyond that, the observed patterns of forgetting in class-incremental semantic segmentation seem to differ from those reported in previous studies that addressed class-incremental classification [Mas20, Hsu18]. This chapter analyzes how catastrophic forgetting affects a semantic segmentation model in class-incremental and domain-incremental learning with the aim of gaining a deeper understanding of these findings.

Therefore, Sec. 6.1 investigates the principles of catastrophic forgetting for class-incremental semantic segmentation (CiSS), answering how the background class affects catastrophic forgetting, why knowledge distillation is more effective in semantic segmentation and where the models' representations are most affected by activation drift. To understand how the principles of forgetting between domain- and class-incremental learning differ, Sec. 6.2 discusses the causes and effects of catastrophic forgetting in domain-incremental learning, answering how low-level feature reuse, batch normalization and pre-training can affect domain-incremental learning. The final section of the chapter illustrates the differences between catastrophic forgetting in class- and domain-incremental learning and discusses the implications for continual learning. The presented results in this chapter are mainly based on two of the author's publications [Kal22a, Kal23c].

## 6.1 Principles of Catastrophic Forgetting in Class-Incremental Learning

When comparing the most successful approaches in class-incremental image classification [Mas20] and CiSS, which were introduced in Sec. 3.2.1, it is noticeable that contrary to classification in CiSS, most of the recent methods build on the idea of Learning without Forgetting (LwF) [Li18] and utilize a knowledge distillation-based loss [Hin15]. The findings in Ch. 4 suggest that knowledge distillation-based approaches even outperform replay-based methods in CiSS. However, they require previously learned classes reappear in future training images because otherwise they would still suffer from forgetting, as the experiments in this section will further confirm. The two main challenges in CiSS that have to be overcome are: (1) catastrophic forgetting of old classes and (2) the semantic shift of the background class. Although existing methods significantly reduce catastrophic forgetting and help to overcome these challenges, there is limited understanding of how the drop in accuracy manifests itself within the deep neural network.

The focus of this chapter is to identify the causes and effects of forgetting that arise in the neural network in CiSS. Specifically, it aims at revealing how activation drift, inter-task confusion and task-recency bias affect the performance in CiSS and how existing approaches overcome these effects. Therefore, Sec. 6.1.2 studies the impact of the semantic shift of the background class on PascalVoc-2012 [Eve12] in three different task protocols with varying degrees of semantic shift of the background class, namely: *overlapped*, *disjoint* and a novel *full disjoint* setup. Next, Sec. 6.1.3 analyzes the degree of activation drift in various layers by stitching them with the previous task's network, as described in Sec. 5.3.1. Using layer stitching and the three task protocols, it is revealed that the semantic shift of the background class is the main cause of the catastrophic drop in performance in CiSS and that forgetting mainly occurs in the decoder layers of the model, where discriminating features for old classes of the encoder are assigned to new visually similar classes or to the background class. However, at the same time, the re-appearance of previous classes in the background of subsequent training tasks also reduces the

internal activation drift in the encoder. Furthermore, decoder retraining is utilized to analyze the degree to which the decoder of the network contributes to inter-task confusion. It is observed that methods that do not use any form of replay fail to learn discriminating features for all classes. Specifically, the model is not able to distinguish old classes from new classes that are visually closely related, e.g., *train* and *bus*. Finally, the results are also extended to varying CNN architectures to support the previous inferences.

### 6.1.1 Experimental Setup

The experiments in this chapter are conducted on the PascalVoc-2012 [Eve12] dataset, which contains 20 object classes and one background class. The experiment setup follows the established PascalVoc-15-5 split that is widely used in CiSS [Dou21a, Mic19, Mic21, Cer20]. The PascalVOC-15-5 split is a two-step incremental learning task that consists of learning 15 classes (1–15) in the first step  $T_0$  and the remaining 5 classes (16–20) in the second step  $T_1$ . The first two training setups follow the *disjoint* and *overlapped* settings proposed by Cermelli et al. [Cer20]. In both settings, only the set of current classes  $\mathcal{S}_k$  is labeled, while the rest is labeled as background  $b$ . However, in the *disjoint* setting, the images of the current task  $T_k$  only contain pixels of classes  $\mathcal{C}_k$ , meaning that images that contain pixels belonging to classes of future tasks will be discarded in the training set of  $T_k$ . In the *overlapped* setting, pixels can belong to any of the classes, but classes that do not belong to the current training set  $\mathcal{S}_k$  will be labeled as background.

Finally, an additional *full disjoint* setting is proposed to examine the effect of the semantic shift that the background class is subjected to in the *disjoint* and *overlapped* settings. The *full disjoint* setting completely avoids the semantic background shift, as each task only contains pixels belonging to the current set of classes, so that classes from the other task will not appear in the background of the current training set. However, it should be emphasized that this scenario is highly unlikely in practice because in semantic segmentation classes would naturally reappear in subsequent tasks.

## ***Models***

Similarly as in the previous chapters, ERFNet [Rom18] is used in the evaluation. The reason is that the underlying effects of forgetting are similar to more established models like DeepLabV3+ [Che18], but at the same time ERFNet is more susceptible to forgetting due to its smaller size, which exaggerates the effect of the causes of forgetting. The findings in this section are confirmed in Sec. 6.1.6 with DeepLabV3+, U-Net [Ron15], BiSeNet V2 [Yu21] and HRNetV2-W48 [Wan19]. The models in the following experiments are initialized with the same randomly initialized weights for every method, as pre-training can increase robustness to catastrophic forgetting [Gal21, Meh21].

## ***Optimization Strategy***

ERFNet is optimized with the SGD optimizer with a weight decay factor of  $3 \times 10^{-4}$ , momentum of 0.9 and an initial learning rate of 0.07 for the first task and  $5 \times 10^{-4}$  for the second task. The learning rate is divided by 2 if the validation loss does not reduce for 8 consecutive epochs. The models are trained for 100 epochs in each task, 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, the data is augmented with the augmentation scheme proposed by Cermelli et al. [Cer20].<sup>1</sup>

## ***Methods Compared***

In this evaluation, representative continual learning approaches from the main categories are compared, including naive fine-tuning approaches, representative regularization and replay methods. For prior regularization methods, EWC [Kir15] and MAS [Alj18] are considered. For data regularization methods, LwF [Li18] is used with the modification of Klingner et al. [Kli20], in which the distillation loss is only applied to the parts of the image

---

<sup>1</sup> Detailed information about implementations, pre-training and the used augmentations can be found in Appendix B.

that are labeled as background. Incremental improvements of knowledge distillation-based approaches [Dou21a, Cer20, Mic21, Mar21] are explicitly left out, as this chapter focuses on evaluating the underlying causes of forgetting. Finally, for replay 20 samples for each class are stored. For every experiment, the results also include an *offline* model, which is jointly trained on all classes in one step. Further information regarding the implementation details and selected hyperparameters can be found in Appendix B.

### 6.1.2 Semantic Background Shift and Class Confusion

This experiment examines the effect of semantic background shift on forgetting through a comparison of selected CiSS methods on the *overlapped*, *disjoint*, and *full disjoint* tasks. As these tasks have a varying degree of semantic shift of the background class, this setup enables to measure the influence that the semantic background shift has on catastrophic forgetting. The results are displayed in Tab. 6.1.

**Table 6.1:** Results of semantic segmentation on Pascal-VOC 2012 in mIoU (%) on the *overlapped*, *disjoint* and *full disjoint* settings. The models are evaluated after training on the complete task sequences.

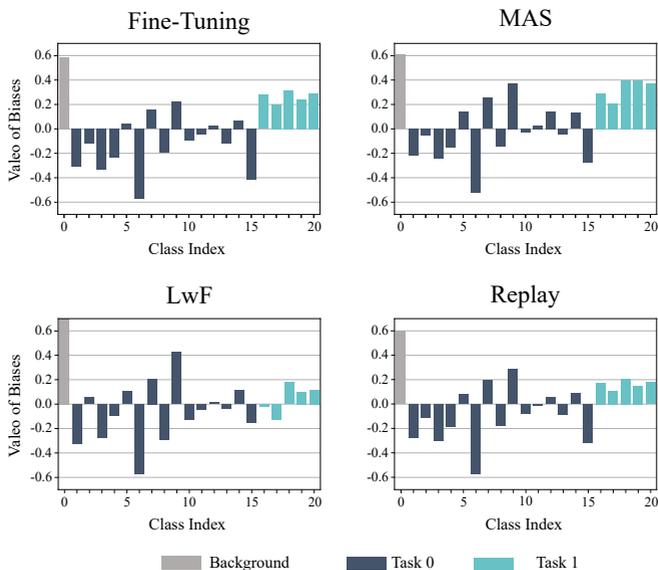
Method	Overlapped			Disjoint			Full Disjoint		
	0–15	16–20	all	0–15	16–20	all	0 - 15	16 - 20	all
Fine-Tuning	4.5	22.2	8.8	4.6	23.0	9.0	5.1	16.3	7.8
MAS [Alj18]	24.1	10.8	21.0	30.6	12.9	26.4	35.6	12.9	30.2
EWC [Kir15]	23.8	11.8	21.0	28.1	10.1	23.8	35.2	10.9	29.4
Replay	41.3	31.5	39.0	42.2	29.1	39.1	48.2	28.8	43.6
LwF [Li18]	45.8	28.2	41.6	44.4	25.4	39.9	35.9	12.6	30.4
Offline	55.7	47.6	53.8	55.7	47.6	53.8	55.7	47.6	53.8

For the *overlapped* and *disjoint* tasks, it can be noted that only LwF and replay effectively learn to discriminate between all classes. EWC and MAS effectively mitigate the forgetting of old classes (0–15) compared to fine-tuning, but they also inhibit the learning of new classes (16–20). The reason for the low mIoU for EWC and MAS on all classes can be inferred from the confusion matrix in Figs. 6.2b and 6.2c, in which EWC and MAS exhibit a strong bias

to the background class and a minor bias towards a few new classes, which is also visible in the bias values of the final convolutional layer in Fig. 6.1. LwF and replay notably reduce both biases. In the *overlapped* and *disjoint* settings, LwF and replay achieve similar performance, as in this setting LwF can effectively replay old classes by discovering them in the background of new images. However, once these classes do not reappear in the background, as is the case in the *full disjoint* task, LwF develops a strong bias towards selected new classes. In contrast to this, replay benefits from the *full disjoint* setting as the training is no longer affected by the semantic background shift. Similarly, MAS and EWC also show significant improvement in this setting, as they benefit from the fact that old classes do not appear as background in the new task, thus not interfering with previously learned knowledge. This is especially noticeable in the confusion matrices of the *full disjoint* setting in Figs. 6.2h and 6.2i, in which the bias towards the background class is greatly reduced<sup>1</sup>.

---

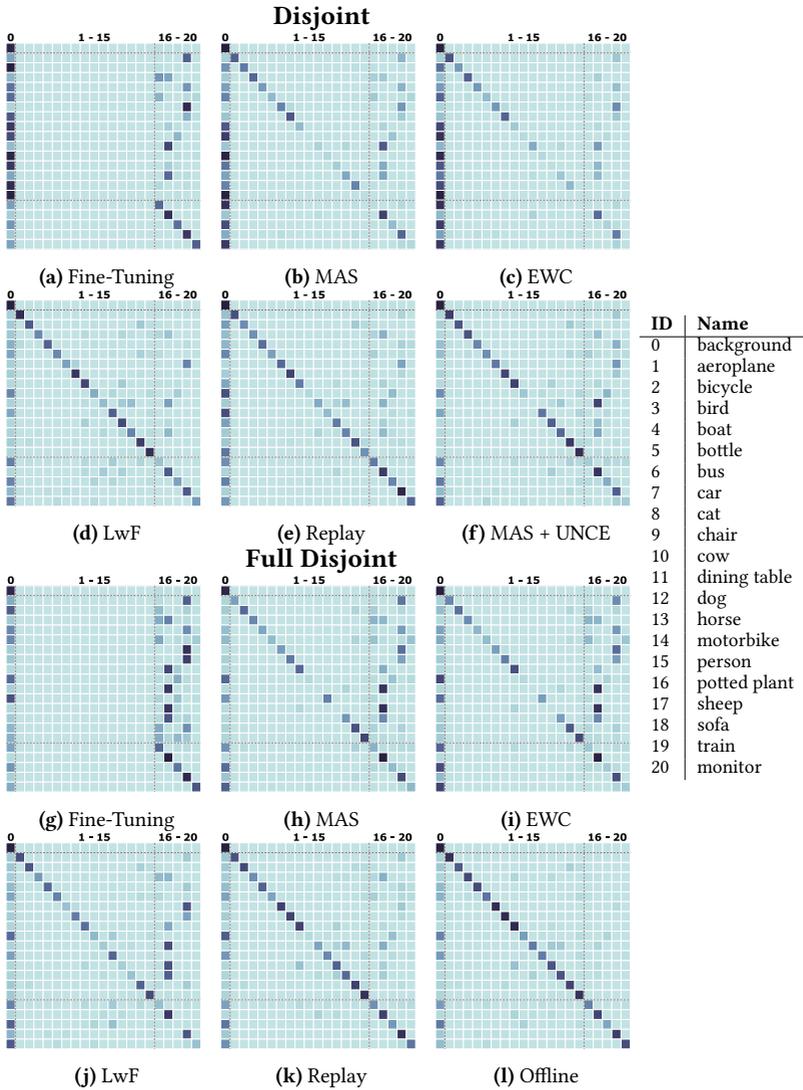
<sup>1</sup> The confusion matrices for the *overlapped* setting are displayed in Appendix A.



**Figure 6.1:** The bias values of the final convolutional layer after training on the *disjoint* PascalVoc-15-5 split show severe bias towards the most recent classes 15–20 and the background class.

This demonstrates that the semantic shift of the background class is a significant cause of forgetting for prior regularization methods like EWC and MAS and that it has a noticeable effect on replay as well. The results also indicate that knowledge distillation is the most effective method to combat this semantic shift.

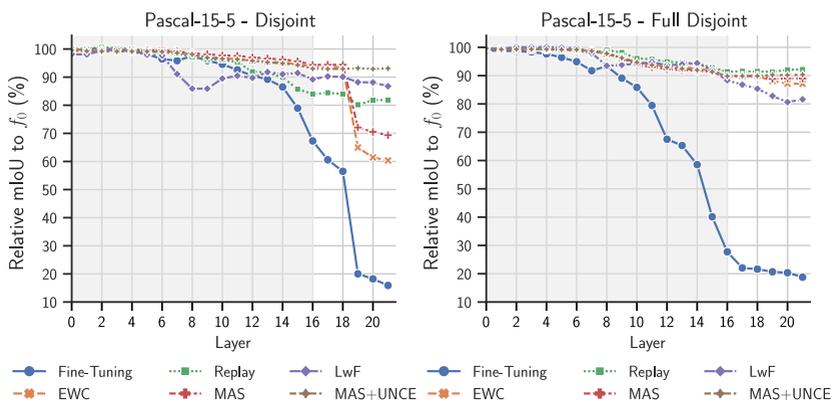
Finally, upon a closer look at the semantics of the false positives in the confusion matrices, it is apparent that old classes that are falsely assigned to a new class share semantic and visual properties. In this case, *bus* (6), *car* (7), *boat* (5) are assigned to *train* (19), whereas *cow* (10) and *horse* (13) are classified as *sheep* (17). The remaining classes that do not share such a relationship with the new classes are falsely classified as background. This confusion can only be alleviated by either replay or LwF when old classes reappear in the background in subsequent tasks.



**Figure 6.2:** Confusion matrices after training on PascalVoc-15-5 *disjoint*. The confusion matrix for (a) Fine-tuning shows a severe bias to the background class and the classes of the most recent task (16–20). EWC [Kir15] and MAS [Alj18] decrease the bias in exchange for worse accuracy on the most recent classes. Replay and LwF [Li18] reduce the bias towards new classes and the background.

### 6.1.3 Activation Drift in Class-Incremental Learning

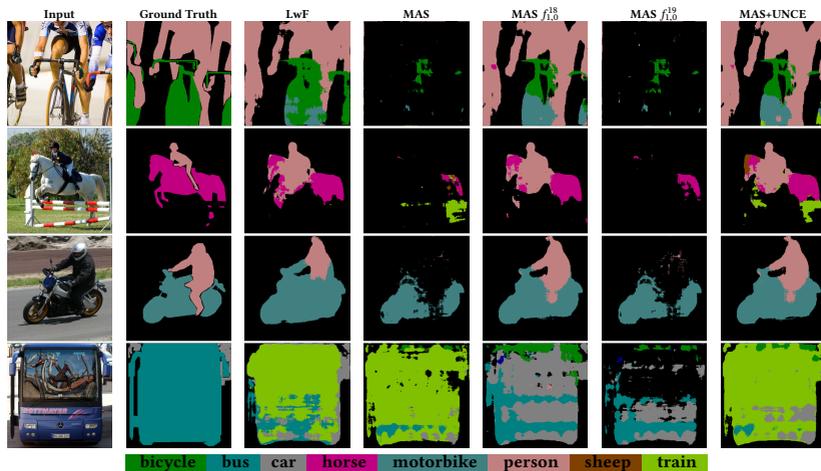
To clarify which layers of the model contribute the most to the bias for the background class and newly learned classes, the internal activation drift is measured using layer stitching. Specifically, the activation drift for each layer is measured between the model before and after learning Task 1, in order to identify the layers most affected by internal activation drift. The setup without an additional stitching layer is used, in which the activations of the model  $f_1$  at the layer  $n$  under examination are directly propagated to the subsequent layer  $n + 1$  in the model  $f_0$ , resulting in the stitched network  $f_{1,0}^n$ . For each stitched network  $f_{1,0}^n$ , Fig. 6.3 displays the mIoU relative to the initial performance on the first task.



**Figure 6.3:** Activation drift between  $f_1$  to  $f_0$  measured by relative mIoU on the first task of the networks stitched together at specific layers (horizontal axis). The layers of the encoder are layer 0–15 (grey area), the decoder layers are 16–21 (white area). The activations in the early layers of the encoder stay very stable for all methods, whereas EWC, MAS and fine-tuning have a severe drift in activations in the decoder layers of the network, which is clear evidence that forgetting is mostly affecting later layers in the *disjoint* setting.

First of all, it can be observed that the early layers of the network (layers 0–4) remain relatively stable across all methods, while later layers, particularly those in the decoder, are more vulnerable to activation drift. This finding aligns with prior work on image classification [Dav22, Ney20], indicating that catastrophic forgetting in CiSS is predominantly observed in deeper layers of the network. Moreover, the analysis reveals that EWC and MAS are effective at mitigating the impact of activation drift on deeper layers of the encoder, with their performance dropping to approximately 90 % of the initial mIoU on the *disjoint* task. In contrast, the stitched fine-tuning model suffers a much steeper drop in performance, reaching only 30 % of the initial mIoU. This suggests that forgetting for EWC and MAS is less severe than accuracy in Tab. 6.1 would reveal. The reason for this could be two-fold: Firstly, the bad accuracy could be attributed to the classifier being biased towards new classes (task-recency bias) or secondly, that the regularization methods fail to learn meaningful features that help to discriminate between old and new classes as they are never trained jointly. While a biased classifier is fixed more easily, inter-task confusion is a fundamental shortcoming of prior regularization methods [Les20].

Another striking phenomenon is the severe change of activations at the third decoder layer (layer 17) that fine-tuning, MAS and EWC exhibit on the *disjoint* task. The predictions of the specific stitched networks  $f_{1,0}^{18}$  and  $f_{1,0}^{19}$  in Fig. 6.4, show that  $f_{1,0}^{18}$  is able to correctly classify old classes (*bike*, *person*) as such, but  $f_{1,0}^{18}$  assigns the background class to these regions. Therefore, it can be concluded that the sudden activation change of MAS originates from the fact that features that were evidence for old classes in  $f_0$  are now attributed as evidence for the background class. This validates that the semantic shift of the background class is predominantly affecting the later layers of the decoder and that discriminating features for old classes are, in fact, not forgotten but assigned to the background class. When completely avoiding the semantic background shift in the *full disjoint* task, the activation drift for the fine-tuned model is much more pronounced in the middle layers of the encoder (layer 8–15), which implies that the re-appearance of old classes, even though they are labeled as background, is mitigating the activation drift in the earlier layers of the model.



**Figure 6.4:** Visualizations of the segmentation maps for LwF, MAS and the resulting networks of MAS  $f_{1,0}^{18}$  and  $f_{1,0}^{19}$ . The predictions of  $f_{1,0}^{18}$  and  $f_{1,0}^{19}$  show that up until layer 18 the information for previously learned classes *person* and *horse* is still available, but is assigned to the *background* in layer 18.

### 6.1.4 The Impact of Inter-task Confusion on the Encoder

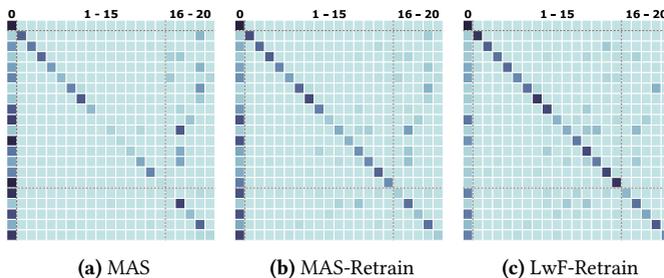
As Sec. 6.1.3 demonstrated that the early layers of a model trained with a continual learning method do not suffer from severe activation drift, the following experiment investigates how useful the learned features of the encoder of the different methods are to discriminate between all classes. Therefore, decoder retrain accuracy is measured by retraining only the decoder of the model on all classes and subsequently evaluating the retrained model on the test set. The first observation to be made when looking at the retraining accuracy in Tab. 6.2, is that all methods improve after decoder retraining, though EWC, MAS and fine-tuning show bigger improvements than LwF and replay. This again confirms that forgetting in the encoder is not as severe for fine-tuning, EWC and MAS as the accuracy indicates. Furthermore, it also verifies that MAS and EWC are effectively preserving important features for old classes

in the encoder, but that the biased decoder layer might wrongly attribute important features for old classes to the background class or new classes, which leads to a severe amount of misclassifications.

**Table 6.2:** Decoder retraining results on Pascal-VOC,  $mIoU_I$  and  $mIoU_R$  denote the  $mIoU$  (%) before and after retraining and  $\Delta$  the discrepancy between  $mIoU_I$  and  $mIoU_R$ .

Method	Overlapped			Disjoint			Full Disjoint		
	$mIoU_I \uparrow$	$mIoU_R \uparrow$	$\Delta \downarrow$	$mIoU_I \uparrow$	$mIoU_R \uparrow$	$\Delta \downarrow$	$mIoU_I \uparrow$	$mIoU_R \uparrow$	$\Delta \downarrow$
Fine-Tuning	8.8	28.0	19.2	9.0	27.9	18.9	7.8	22.0	14.2
MAS [Alj18]	21.0	34.3	13.3	26.4	36.2	9.8	30.2	37.3	7.1
EWC [Kir15]	21.0	34.3	13.3	23.8	35.1	11.3	29.4	36.9	7.5
LwF [Li18]	41.6	45.3	3.7	39.9	43.3	3.4	30.4	38.1	7.7
Replay	39.0	42.6	3.6	39.1	42.9	3.8	43.6	45.6	2.0
Offline	53.8	54.6	0.8	53.8	54.6	0.8	53.8	54.6	0.8

Still, as EWC, MAS and fine-tuning do not achieve a comparable  $mIoU$  as LwF or replay after decoder retraining, it can be concluded that the learned features of the encoder are less useful for discriminating between all classes. Specifically, the aforementioned related classes *bus* (6), *car* (7), *boat* (5), *train* (19), as well as *cow* (10), *horse* (13), *sheep* (17) cannot be effectively classified after retraining, compare Fig. 6.5. This indicates that replay does not suffer from inter-task confusion since old classes are taken into account when optimizing for new classes, leading to more discriminative features. The same holds for LwF in the *overlapped* and *disjoint* setting, in which old classes are effectively replayed by using soft-labels for old classes that are discovered in the background.



**Figure 6.5:** Confusion matrices before (a) and after (b), (c) retraining the decoder on all classes of PascalVoc2012.

### 6.1.5 Reducing Background Bias and Task Recency Bias

A simple method to reduce the recency bias in the classification layer that is used in class-incremental classification is to calculate the cross entropy loss (CE) only for classes of the current training set [Mas20]. This enforces that errors are only back-propagated for probabilities that are related to the current set of classes  $\mathcal{S}_k$  instead of all classes that were already observed  $\mathcal{C}_k$ :

$$\mathcal{L}_{\text{ce}}(\mathcal{Y}, \hat{\mathcal{Y}}) = -\frac{1}{|\mathcal{I}|} \sum_{i \in \mathcal{I}} \sum_{c \in \mathcal{S}_k} y_{i,c} \log(\hat{y}_{i,c}) \quad (6.1)$$

However, in the case of CiSS, this addition has proven to be less effective than the standard cross-entropy loss [Dou21b]. Therefore, an unbiased cross entropy loss (UNCE) is proposed by Cermelli et al. [Cer20], which accounts for the uncertainty of the content of the background class. This is achieved by comparing the pixels that are labeled as background with the probability of having either an old class or the background predicted by the model:

$$\mathcal{L}_{\text{UNCE}}(\mathcal{Y}, \hat{\mathcal{Y}}) = -\frac{1}{|\mathcal{I}|} \sum_{i \in \mathcal{I}} \sum_{c \in \mathcal{C}_k} y_{i,c} \log(\hat{y}_{i,c}) \quad (6.2)$$

$$\hat{y}_{i,c} = \begin{cases} \sum_{q \in \mathcal{C}_{k-1}} \hat{y}_{i,q} & \text{if } c = c_b \\ \hat{y}_{i,c} & \text{otherwise} \end{cases} \quad (6.3)$$

The variable  $c_b$  denotes the index of the background class in the class set of  $\mathcal{C}_k$ . In addition, weight normalization layers [Sal16] were also successfully used in classification tasks to address the recency bias [Les21].

The results in Tab. 6.3 show the impact of UNCE and UNCE combined with weight normalization to combat the recency and background bias in CiSS. Overall, UNCE improves the accuracy for all approaches on the *disjoint* setting. Specifically, the prior regularization methods (MAS and EWC) show significantly higher accuracy compared to the basic cross-entropy loss. This can be attributed to the fact that UNCE effectively mitigates the background bias, as can be seen in the confusion matrix in Fig. 6.2f and the segmentation maps in Fig. 6.4. In addition, the severe activation drift that is observed in Fig. 6.3 between layers 18 and 19 for MAS completely vanishes with the use of UNCE. Therefore, UNCE effectively resolves the confusion between the old classes and the background class. This confirms the assumption that a major cause of forgetting was in fact a bias of the classifier towards the background and the new classes. However, the confusion matrices show that while the background bias is severely reduced by using UNCE, the semantic confusion of old and new classes is amplified.

In the *full disjoint* setting, the use of UNCE does not improve the performance as much as it does in the *disjoint* setting. The reason is that in the *full disjoint* setting the pixels of old classes do not re-occur and thus the de-biasing effect of UNCE is decreased. Therefore, the content of the background class plays an important role in mitigating forgetting.

Of the selected approaches, only replay benefits from the addition of the weight normalization layer. Finally, it should be noted that EWC and MAS, with the addition of UNCE, show competitive performance compared to the remaining approaches and more recent approaches like MiB [Cer20], even without the use of knowledge distillation or replay. However, it is likely that

for longer task sequences and more classes, MiB will outperform prior regularization methods, as they will not be able to learn discriminative features.

**Table 6.3:** Results on Pascal-15-5 in mIoU (%) on the *disjoint* and *full disjoint* settings with: cross-entropy loss (CE), unbiased cross-entropy loss (UNCE) and UNCE combined with a weight normalization (UNCE+WN). UNCE effectively reduces forgetting for all approaches, especially for EWC and MAS.

Method	Disjoint UNCE						Full Disjoint UNCE						UNCE+WN		
	0-15	16-20	all	0-15	16-20	all	0-15	16-20	all	0-15	16-20	all	0-15	16-20	all
Fine-Tuning	4.6	23.0	9.0	10.4	21.8	13.1	16.5	21.6	17.7	5.1	16.3	7.8	6.0	15.5	8.3
EWC [Kir15]	28.1	10.1	23.8	48.2	11.6	39.4	17.0	9.4	15.2	35.2	10.9	29.4	41.1	9.8	33.6
MAS [Alj18]	30.6	12.9	26.4	45.8	14.4	38.3	41.0	13.9	34.6	35.6	12.9	30.2	39.1	12.3	32.7
LwF [Lil18]	44.4	25.4	39.9	45.3	22.9	40.0	46.6	19.7	40.2	35.9	12.6	30.4	38.0	13.8	32.2
Replay	42.2	29.1	39.1	47.2	31.4	43.5	48.1	31.9	44.3	48.2	28.8	43.6	47.7	28.0	43.0
MiB [Cer20]	-	-	-	48.6	21.7	42.2	49.4	24.1	43.3	-	-	-	47.6	19.7	41.0

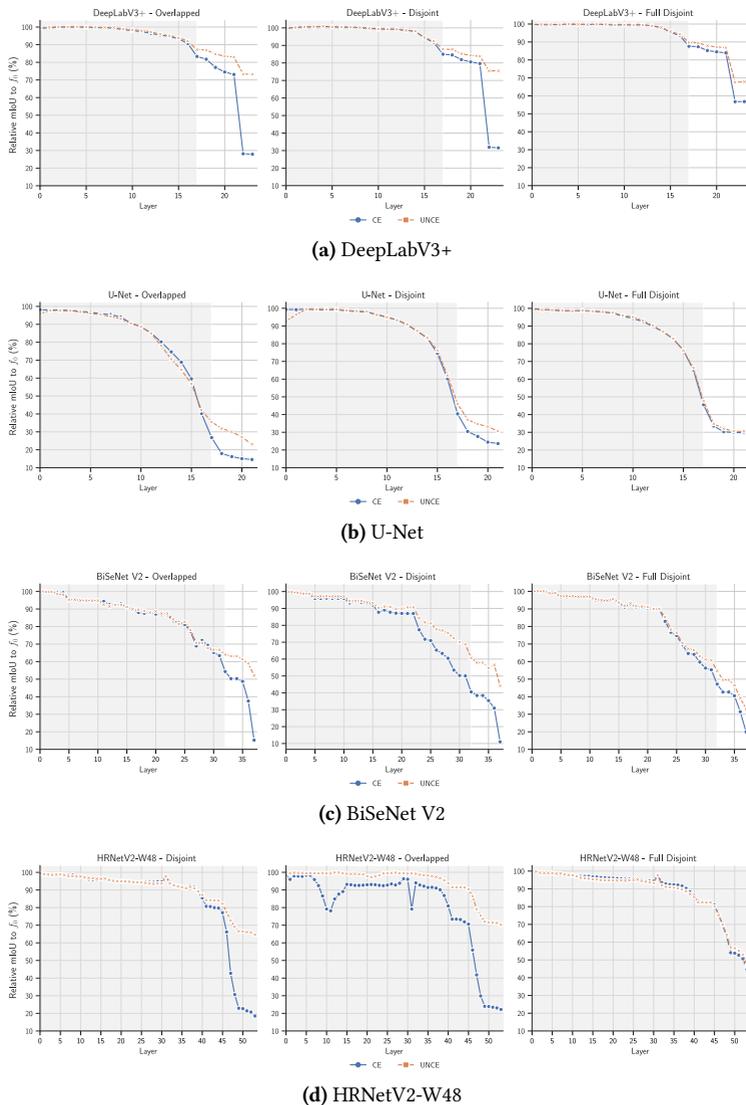
## 6.1.6 Effects of Different Architectures

Finally, the findings so far are confirmed on different CNN architectures, namely DeepLabV3+, U-Net, HRNetV2-W48 and BiSeNet V2. These specific architectures are chosen to verify whether even deeper models with more parameters are affected by the same effects and, additionally, how models that have multi-stage input for the decoder are affected in class-incremental learning, such as U-Net. Similar to before, the models are evaluated on all three settings: *overlapped*, *disjoint* and *full disjoint*. The models are trained incrementally using fine-tuning, either with the basic cross-entropy or the UNCE loss.

The results in Tab. 6.4 overall confirm the previous findings. Using the basic CE loss leads to severe forgetting of classes 0–15, whereas introducing the UNCE loss significantly reduces forgetting in the *overlapped* and *disjoint* settings as classes reappear in the second task’s images. However, it is striking that when using the CE loss, the results for bigger models, such as DeepLabV3+ and HRNetV2-W48, are similar to the smaller ERFNet and BiSeNet V2. Only when using the UNCE loss do the bigger models achieve significantly better results. This occurs because the models are severely biased towards the background class and new classes, so that old classes are almost never correctly identified. Though U-Net, which uses the same ResNet-50

backbone as DeepLabV3+ in these experiments, is significantly more affected by forgetting. This effect can be attributed to the fact that the U-Net decoder receives input from multiple earlier stages of the network, so that during training, early network layers are more susceptible to activation drift.

Indeed, the layer stitching plots in Fig. 6.6 demonstrate that, despite the fact that DeepLabV3+ and U-Net use the same encoder, the early layers of the encoder in U-Net are significantly more affected by activation drift than the corresponding layers in DeepLabV3+. This suggests that the multi-stage input exacerbates the effects of catastrophic forgetting by increasing activation drift in earlier layers. The remaining layer stitching plots of HRNetV2-W48, DeepLabV3+ and BiSeNet V2 confirm the findings for ERFNet.



**Figure 6.6:** Activation drift for different architectures on Pascal-15-5 using fine-tuning with cross-entropy (CE) and unbiased cross-entropy (UNCE). Other CNNs are affected by the same sudden activation drift in layers close to the output layer, specifically in the *overlapped* and *disjoint* setting. Models that have input from multiple encoder stages to the decoder (b) and (c) show increased activation drift in early layers.

**Table 6.4:** Results in mIoU (%) of fine-tuning for the *overlapped*, *disjoint* and *full disjoint* settings using fine-tuning with cross-entropy (CE) and unbiased cross-entropy (UNCE).

Architecture	Overlapped						Disjoint						Full Disjoint					
	CE		UNCE		all		CE		UNCE		all		CE		UNCE		all	
	0-15	16-20	all	0-15	16-20	all	0-15	16-20	all	0-15	16-20	all	0-15	16-20	all	0-15	16-20	all
ERFNet	4.6	23.0	9.0	10.4	21.8	13.1	5.1	16.3	7.8	6.0	15.5	8.3	5.1	16.3	7.8	6.0	15.5	8.3
DeepLabV3+	4.6	25.8	9.6	34.2	24.8	32.0	4.7	26.4	9.8	32.0	25.6	30.5	5.6	19.1	8.8	14.1	18.8	15.3
U-Net	4.7	21.8	8.8	10.5	21.8	13.2	4.5	24.5	9.3	14.0	24.0	16.4	5.3	14.2	7.4	5.8	14.3	7.8
BiSeNet V2	4.5	23.5	9.0	25.7	23.3	25.1	4.5	24.3	9.3	21.5	23.2	21.9	5.3	16.7	8.0	8.1	17.2	10.3
HRNetV2-W48	4.5	25.1	9.4	35.0	23.5	32.3	4.6	24.7	9.4	27.1	22.1	25.9	5.4	15.7	7.8	12.0	17.4	13.3

### 6.1.7 The Role of the Background Class to Overcome Forgetting

The prior observations show that in CiSS the semantic shift of the background class is a major cause of a rapid drop in performance if not addressed correctly. However, if the uncertainty of the content of the background class is taken into account by either UNCE, knowledge distillation or both, the appearance of old classes in the background can to some extent be used for replay. The experiments in the *full disjoint* setting highlight that once classes do not re-occur, these methods are less effective, whereas explicit replay of classes benefits from avoiding the semantic shift.

The ranking of the methods of *full disjoint* setting in CiSS is also similar to the ranking of the same methods for class-incremental image classification (compare Tab. 6.5), indicating that the discrepancy in performance between LwF and replay in image classification is due to the missing background class. Looking at it the other way around, this could also mean that introducing an out-of-set class for image classification could help to reduce forgetting in the class-incremental setting without requiring explicit replay via stored samples, as the re-appearing classes in the out-of-set class play a similar role as explicit replay.

**Table 6.5:** Classification results on PascalVoc-15-5 in (%).

Method	Full Disjoint		
	0 - 15	16 - 20	all
Fine-Tuning	13.6	27.6	17.1
MAS [Alj18]	32.0	27.8	31.0
EWC [Kir15]	27.2	25.4	26.8
Replay	39.6	32.7	37.9
LwF [Li18]	42.1	34.2	40.1
Offline	51.3	54.8	52.2

### 6.1.8 Conclusion

This section highlighted how catastrophic forgetting manifested itself in the hidden representations of the network. It demonstrated that forgetting is concentrated at deeper layers in the decoder, where features that were evidence for old classes are reassigned to the background class or to new, visually similar classes. The experiments with varying degrees of the semantic background shift demonstrate that the background class causes severe forgetting but can also be leveraged to reduce activation drift in the model by using knowledge distillation or a loss that accounts for the uncertainty of the background class. Finally, it was found that only methods that in some form rehearse old classes during training of new classes can learn to correctly discriminate between all classes after incremental training, as otherwise the model fails to learn to discriminate between new and old classes that share similar visual features.

## 6.2 Principles of Catastrophic Forgetting in Domain-Incremental Learning

The previous section examined the causes of forgetting in class-incremental semantic segmentation and explored the impact of semantic shift of the background class on the model’s internal representations. It was found that catastrophic forgetting in class-incremental learning predominantly stems from a pronounced activation drift in the later decoder layers of the network, while the earlier layers remain relatively unaffected. These observations suggest

that, during class increments, the model primarily reassigns features that were previously indicative of old classes to new classes or the background.

However, in the domain-incremental learning setting, the classes remain unchanged while the input distribution undergoes a shift. As a result, it is anticipated that forgetting in domain-incremental semantic segmentation would primarily arise from activation drift in the earlier layers of the network. This chapter aims to validate this assumption by examining how the internal representations of semantic segmentation models are affected during domain-incremental learning and with that to identify the specific effects and causes of catastrophic forgetting in domain-incremental semantic segmentation (DiSS).

Furthermore, because transfer learning has been shown to be more effective when previously learned features are reused for the downstream task [Ney20], it is hypothesized that reusing features can also mitigate catastrophic forgetting by preventing overwriting of previous features. To test this hypothesis, this section also investigates the impact of pre-training and various augmentation techniques that should facilitate more general features in the model, with the goal to reduce the internal representation shift in the model. Similarly, this hypothesis indicates that if subsequent tasks are more visually similar to each other, more features may be reused because they likely share similar features, which in turn should reduce catastrophic forgetting. In order to investigate this, the models will be evaluated in four parallel domain-incremental scenarios in which they are adapted to varying adverse weather conditions.

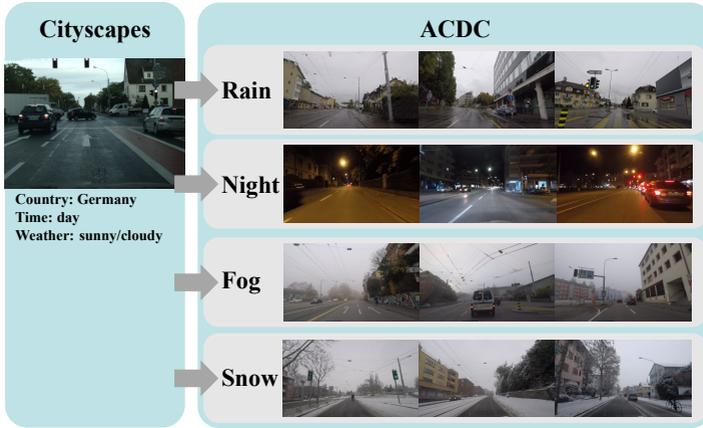
## 6.2.1 Related Work

Recent work by Neyshabur et al. [Ney20] investigates the role of feature reuse in transfer learning, in which they have shown that for a successful transfer, feature reuse and low-level statistics of the data are important. They observe that compared to randomly initialized models, pre-trained models are in the same basins of the loss landscape and develop similar features in the early layers of the model, especially when the downstream task shares similar visual features with the pre-training domain. This work inspired the idea that feature reuse might also be vital to mitigate forgetting. Similarly, Mirzadeh et al.

[Mir20] demonstrated that common regularization techniques, such as dropout and learning rate decay, lead to a reduction of catastrophic forgetting by widening the current task’s local minima, which again links better generalization capabilities to reduced forgetting. Concurrent work also highlights that improved domain generalization can be achieved by utilizing a combination of various augmentations that are used during training on a source domain [Sch23a]. Furthermore, several works investigate the impact of pre-training in continual learning, in which they empirically demonstrate that it reduces forgetting by leading to wider local minima [Meh21] and that self-supervised methods are even more data-efficient than supervised pre-training methods [Gal21]. This section links the ideas of pre-training and improved generalization to increased feature reuse and matching of low-level statistics.

### 6.2.2 Experimental Setup

The experiments in this section are all conducted in a domain-incremental setting, which involves adapting from the Cityscapes (CS) [Cor16] dataset to ACDC [Sak21], which is commonly used as a benchmark for unsupervised domain adaptation in the automated driving domain for adverse weather conditions. As previously stated, the CS dataset is an automotive semantic segmentation dataset collected during daytime and dry weather conditions in different German, Swiss and French cities. 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 the changes between the tasks are only based on the domain differences. To study how features are reused or adapted in each adverse weather condition, four different scenarios are investigated, all starting with the same CS model:  $CS \rightarrow \textit{Night}$ ,  $CS \rightarrow \textit{Rain}$ ,  $CS \rightarrow \textit{Snow}$  and  $CS \rightarrow \textit{Fog}$ . The setup is also illustrated in Fig. 6.7.



**Figure 6.7:** The four different domain-incremental setups all starting from a model trained on Cityscapes and then are either adapted to *Rain*, *Night*, *Fog* or *Snow* subsets.

## Models

In the following experiments, the widely adopted DeepLabV3+ [Che18] with a ResNet-50 backbone is used, as DeepLabV3+ is a commonly used architecture in domain adaptation. Similarly to Sec. 6.1, the findings in this section are confirmed with different architecture in Sec. 6.2.9, because architectural choices can have a significant impact on continual learning [Mir22b]. In the majority of the experiments, the models are initialized with random weights for training on the first task, as pre-training is suspected to increase robustness to catastrophic forgetting [Gal21, Meh21] by enabling low-level feature reuse, as Sec. 6.2.5 will demonstrate.

## Optimization Strategy

The models are optimized with SGD with momentum of 0.9, weight decay of  $3 \times 10^{-3}$  and a batch size of 8. The learning rate is set according to a polynomial learning rate schedule with power 0.9 and starts CS training with

a 0.07 learning rate. The models are trained for 200 epochs on Cityscapes and then fine-tuned for 150 epochs on ACDC. During optimization on ACDC the optimizer starts with a reduced learning rate of  $5 \times 10^{-3}$ . The images are cropped to  $512 \times 1024$  and are randomly flipped and scaled during training. Unless stated otherwise, no other augmentations are used. During testing, the images are used without any scaling or cropping.<sup>1</sup>

### 6.2.3 Activation Drift after Incremental Adaptation

To understand the extent of activation drift that occurs in CNN models when naively adapting to different adverse weather conditions, DeepLabV3+ is first trained on Cityscapes and subsequently fine-tuned individually on each of the four ACDC subsets. The results are displayed in Tab. 6.6.

**Table 6.6:** Results on  $CS \rightarrow ACDC$  in mIoU (%) for each subset of ACDC. While the zero-shot performance for *Night* is the worst, after the fine-tuning to *Night*, it is least affected by forgetting.

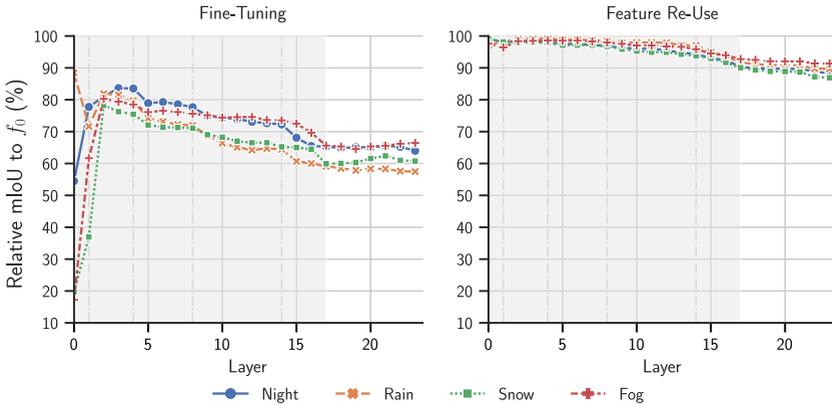
Task 2	Task 1		Task 2		
	mIoU <sub>CS</sub>	mIoU <sub>T2</sub>	mIoU <sub>CS</sub>	mIoU <sub>T2</sub>	forgetting
Rain	72.0	30.4	38.8	57.7	33.2
Night	72.0	10.5	45.9	43.6	26.1
Snow	72.0	23.1	42.2	62.3	29.8
Fog	72.0	33.4	44.0	69.0	28.0

As one would expect, the zero-shot performance on domains that are visually more similar to Cityscapes, such as *Fog* and *Rain*, is significantly better than for dissimilar domains such as *Snow* and *Night*, with *Night*, displaying the lowest performance at only 10.4% mIoU<sub>0,1</sub>. This is explained by the obvious differences between the domains, with the day-to-night shift and snow-covered landscape representing a larger shift from Cityscapes than the wet environment or foggy conditions [Sak21]. However, after fine-tuning the models on

<sup>1</sup> Detailed information about implementations, pre-training and the used augmentations can be found in Appendix C

the adverse subsets, it can be observed that the better zero-shot performance on *Rain* and *Fog* does not indicate less forgetting when compared to *Snow* and *Night*. Most strikingly, forgetting is the lowest after adapting to *Night*.

To understand the cause of this, it is necessary to determine how the individual models in these settings are affected by activation drift and what specific layers are most affected. The activation drift is measured with layer stitching between the model before and after learning the second task. The setup explained in Sec. 5.3.1 is utilized so that the activations of the layer  $n$  under examination  $f_1$  are directly propagated to the layer  $n + 1$  in  $f_0$ . The resulting network  $f_{1,0}^n$  is then evaluated on Cityscapes. The mIoU relative to the initial performance on the first task is illustrated in Fig. 6.8.



**Figure 6.8:** Activation drift between models  $f_1$  to  $f_0$  measured by relative mIoU on the first task of the models 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 dashed grey lines indicate the start of a ResNet block. Layer-stitching reveals that during domain-incremental learning, changes in low-level features are a major cause of forgetting. With an improved training scheme, combining simple augmentations, exchanging normalization layers and using pre-training, the model is optimized to reuse low-level features during incremental learning, leading to significant reduction of catastrophic forgetting.

Remarkably, Fig. 6.8 shows that activation drift predominantly affects the first few layers of the network, which differs from class-incremental learning settings where early layers tend to remain stable, as it was also observed in Sec. 6.1. Specifically, the low-level features of the models tuned to *Fog* and *Snow* cannot be reused by the initial model, indicating significant activation drift in the shallow layers of the network.

However, after this initial drop in relative mIoU, a substantial increase is noticeable after the first ResNet block, suggesting that later features are indeed reused by the Cityscapes model. At that point, it is likely that features are more abstract and, consequently, more useful for the model trained on CS. After the second ResNet block a gradual decline in relative mIoU until the decoder layers is observed.

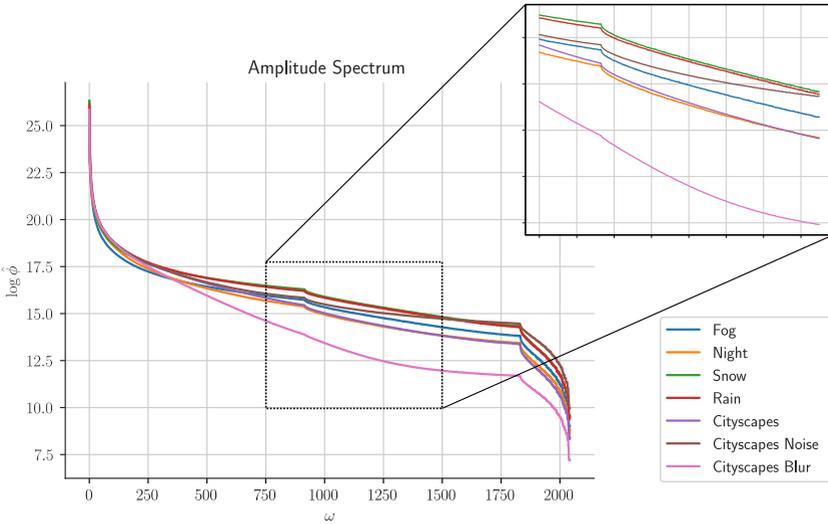
It is important to note that once the representations have shifted, subsequent layers are unlikely to regain similarity, as their representations are based on the output of the previous layer. The changing image distribution is likely responsible for the initial feature disparity observed. Thus, the next section analyzes the pixel-level image distribution in the respective domains.

#### 6.2.4 Analysis of Image Statistics

To explain the significant representation changes observed in the early layers, the domains are first compared based on their corresponding pixel mean and standard deviation for each HSV channel. The results are displayed in Tab. 6.7. It is evident that there are substantial differences between the domains, particularly with *Rain*, *Snow*, and *Fog* being noticeably brighter compared to Cityscapes. These variations in brightness can contribute to shifts in low-level features within the network.

Additionally, as the generalization ability of CNNs can be negatively affected by the exploitation of mid- to high-frequency components in images [Wan20, Abe21], the Fourier amplitude spectra in the frequency domain of the different training tasks are also illustrated in Fig. 6.9. It is observed that the domains

exhibit similarity in the low- to mid-frequency ranges, but *Snow* and *Rain* contain significantly more high-frequency components. This discrepancy in the frequency domain could potentially lead to overfitting on high-frequency features when the model is trained on these domains, consequently amplifying the catastrophic forgetting.



**Figure 6.9:** Fourier 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 high frequency components of the image.

**Table 6.7:** 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.

Dataset	Mean			Standard Deviation		
	Hue	Saturation	Value	Hue	Saturation	Value
Rain	86	32	110	62	35	78
Snow	104	19	132	55	21	62
Night	64	122	60	66	64	46
Fog	93	20	131	60	23	64
Cityscapes	59	49	83	18	22	49

### 6.2.5 Adjusting Low-Level Features

Prior studies have highlighted the importance of low-level feature reuse for successful transfer learning [Ney20] and demonstrated that pre-training can mitigate forgetting [Gal21, Meh21]. The following series of experiments demonstrates that low-level feature reuse not only facilitates knowledge transfer to subsequent tasks but also plays a critical role in preventing catastrophic forgetting.

To validate this claim, the model  $f_0$  is initialized with various pre-training and augmentation protocols on Cityscapes that should facilitate more generalized features, with the expectation that those features will be reused when the model is adapted to the different ACDC subsets, leading to reduced activation drift and decreased catastrophic forgetting. It should be noted that the model is fine-tuned on the ACDC subsets without augmentations or continual learning algorithms.

To examine how augmentations and pre-training impact the model’s feature reuse and reduce forgetting, they are examined separately. For studying the impact of different pre-training strategies, the ResNet-50 model is initialized with weights trained on ImageNet either fully-supervised or using the self-supervised learning (SSL) methods DINO [Car21], MoCo v3 [Che21a], SwAV [Car21] and BarlowTwins [Zbo21]. For the augmentation experiments, the following strategies are employed:

- Using AutoAlbument (AutoAlbum) [Bus20], to learn an image augmentation policy from the CS dataset using Faster AutoAugment [Hat20].
- Learning color-invariant features by intensive color jittering and randomly rearranging input image channels. This combination is denoted 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. 6.9.

Finally, the experiments also include an offline pre-trained model that is trained jointly on the CS and ACDC subsets and then fine-tuned on the

target task. Offline pre-training allows to establish an upper bound on feature reuse, as the model should have learned features that are the joint optimum for both tasks.

**Table 6.8:** Results on  $CS \rightarrow ACDC$  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.

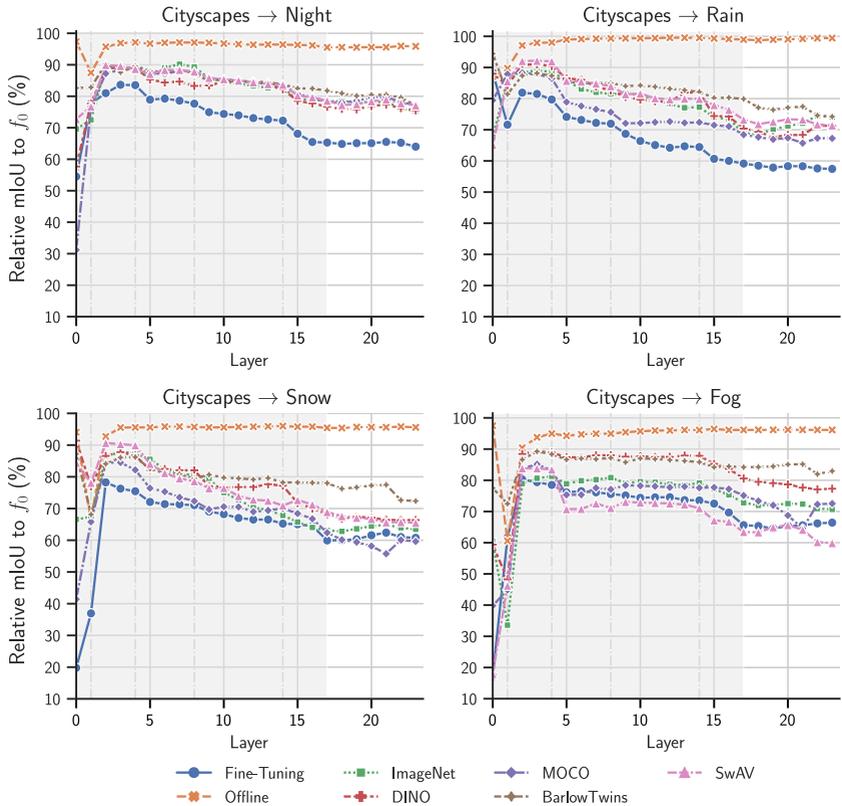
Method	Cityscapes Test mIoU	Night			Rain			Fog			Snow			
		Zero Shot	Test mIoU	forg.										
Augment.	FT	72.0	10.5	43.6	26.1	30.4	57.7	33.2	33.4	69.0	28.0	23.1	62.3	33.2
	AutoAlb.	72.2	24.2	47.3	15.1	42.8	59.4	10.7	50.1	68.2	14.7	37.2	63.8	10.7
	Distort	71.7	19.8	46.5	16.2	38.9	60.9	19.0	46.5	68.3	15.2	32.8	62.3	19.0
	Gaus	69.1	8.1	46.3	23.0	26.9	60.0	26.7	26.9	65.4	28.3	15.8	64.7	26.5
	Noise	69.8	9.6	46.7	21.3	27.8	60.6	25.2	27.5	69.3	30.2	21.3	63.2	30.2
Pre- Training	ImageNet	73.9	6.3	47.5	19.1	27.3	60.9	22.5	21.8	68.8	23.8	25.8	66.2	28.8
	MOCO	75.2	14.0	48.3	17.3	32.6	63.5	26.8	41.9	72.3	22.3	30.3	66.3	33.8
	DINO	75.0	11.6	49.7	18.7	28.2	64.4	23.4	35.2	72.4	19.6	27.3	67.0	27.1
	BarlowT.	73.9	14.3	47.3	17.1	34.8	65.7	22.5	41.6	71.4	14.4	31.8	65.4	22.0
	SwAV	76.4	15.8	48.1	17.8	31.0	62.4	24.5	40.9	71.7	32.8	28.0	66.4	27.8
	Offline	-	46.1	47.4	3.2	59.0	59.1	0.5	67.5	66.9	3.2	62.4	62.9	3.3

The results displayed in Tab. 6.8 suggest that both pre-training and augmentations during CS training can lead to better transfer to subsequent tasks and reduced forgetting on CS. However, while pre-training improves transfer to new tasks, it only moderately enhances zero-shot capabilities compared to the model without pre-training. On the other hand, Distortion and AutoAlbum enhance zero-shot performance but exhibit slightly worse performance on the ACDC tasks compared to the pre-trained models. This finding indicates that better zero-shot performance does not always translate to better transfer performance.

Nevertheless, these augmentations prove to be the most effective in mitigating forgetting for all tasks. Augmentations that focus on removing high-frequency components are less effective in reducing forgetting or, in the case of *Fog*, even lead to a further decrease in mIoU on the previous task. It should be noted that for the domains *Rain* and *Snow*, which contain more high-frequency components than CS, the addition of noise and blurring is more

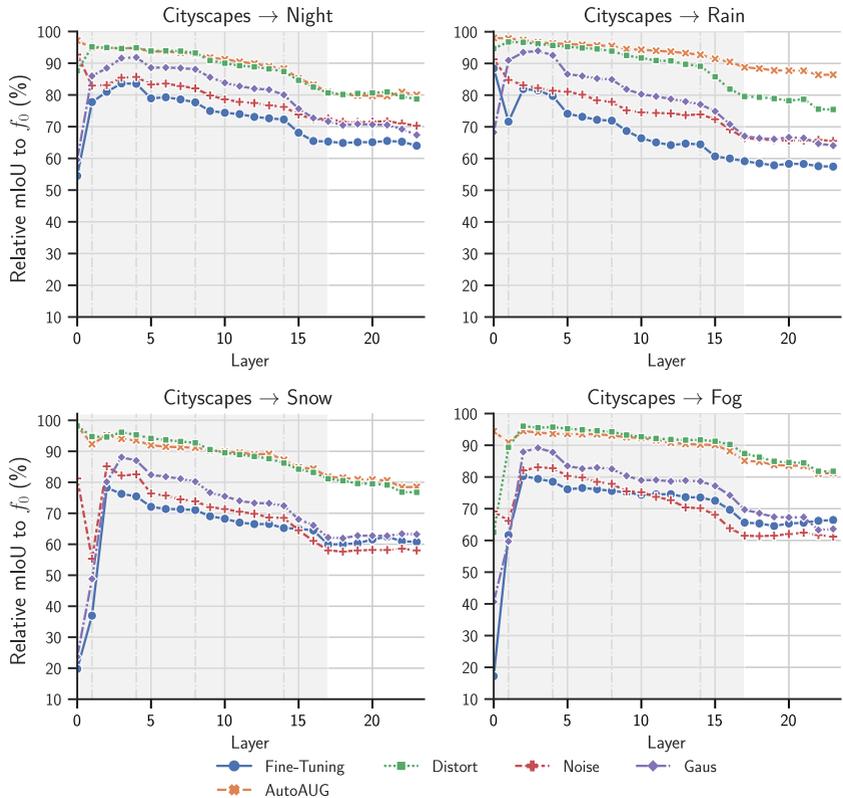
effective than for *Fog* and *Night*.

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.



**Figure 6.10:** The influence of pre-training on the activation drift between  $f_1$  to  $f_0$  measured with layer stitching at specific layers (horizontal axis). The dashed grey lines indicate the start of a ResNet block. The activations up until the first ResNet block undergo drastic changes. After the first block 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  $f_0$ .

To understand the effect on the internal representations, the layer stitching plots in Figs. 6.10 and 6.11 display the activation drift that the models are subjected to. The pre-trained models, displayed in Fig. 6.10, have a reduced initial drop in similarity at the first layer than randomly initialized models (fine-tuning). Furthermore, the similarity of activations in intermediate layers also remains notably higher compared to fine-tuning. Interestingly, even the offline pre-trained model experiences a noticeable drop in similarity at the first layer for *Snow* and *Fog*, as well as a moderate drop for *Rain* and *Fog*. Later experiments will confirm that this is largely due to the biased population mean and standard deviation of the BN layers.



**Figure 6.11:** The influence of augmentations on the activation drift between  $f_1$  to  $f_0$  measured with layer stitching at specific layers (horizontal axis). The dashed grey lines indicate the start of a ResNet block. The activations up until the first ResNet block undergo drastic changes. After the first block 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  $f_0$ .

Most notably, the models trained with color augmentation during training do not encounter this initial drop in similarity, despite undergoing the same optimization process for the second task as the offline and pre-trained models, which can be seen in Fig. 6.11. 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. The domain-invariant features that the models trained with augmentation extract will also reduce the bias in the population statistics of the following batch normalization layer to the most recent task, which is often a cause of catastrophic forgetting [Lom20]. This is the most likely explanation for the improved robustness towards catastrophic forgetting and already indicates that more general features in the early layers will reduce the activation drift in the network and consequently reduce forgetting.

### 6.2.6 Impact of Batch Normalization on Forgetting

The results in Sec. 6.2.5 suggest that the changing BN population statistics are a major cause of early layer representation changes. To verify this, the BN layer population statistics are re-estimated on the combined dataset of CS and the specific ACDC subset without changing any parameters, using the BN re-estimation proposed in Sec. 5.4.1. Afterwards, the updated model is evaluated on the CS dataset and the performance of the re-estimated model is reported as  $mIoU_{BN}$  and the change in  $mIoU$  as  $\Delta mIoU_{BN}$ . The results are displayed in Tab. 6.9.

**Table 6.9:** Performance in  $mIoU$  (%) on CS of the model  $f_1$  after re-estimating all BN layer population statistics. FT is most affected by changing population statistics, while models trained with augmentation are least affected.

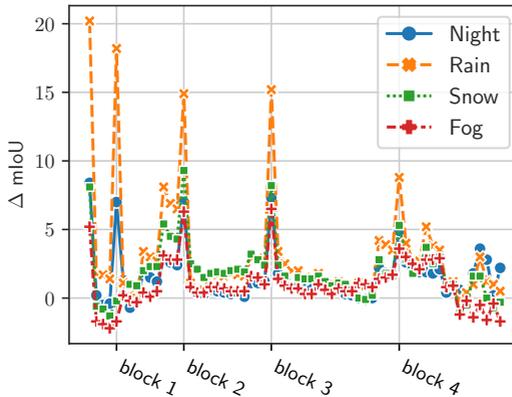
Method	Night		Rain		Fog		Snow	
	$mIoU_R \uparrow$	$\Delta mIoU$	$mIoU_{BN} \uparrow$	$\Delta mIoU_{BN}$	$mIoU_{BN} \uparrow$	$\Delta mIoU_{BN}$	$mIoU_{BN} \uparrow$	$\Delta mIoU_{BN}$
FT	58.6	12.7	58.2	19.4	49.7	5.8	51.3	9.1
AutoAlb.	59.8	2.7	62.0	0.5	54.4	-3.1	53.9	-2.1
Distort	59.2	3.7	59.1	6.5	52.0	-4.5	54.0	0.0
ImageNet	61.7	7.4	59.6	8.1	51.7	2.1	54.7	9.7
MOCO	63.5	5.6	62.5	14.0	55.1	2.2	57.4	16.0
DINO	64.0	7.7	63.8	12.2	57.8	2.4	61.9	14.0
BarlowT	65.4	8.6	65.2	13.9	60.6	8.7	59.4	0.0
Img+Dis	62.5	4.3	63.0	-0.6	55.3	-4.4	59.0	0.3
Offline	68.3	-0.2	70.3	-0.8	68.8	-0.4	69.2	0.4

The results demonstrate that most methods benefit significantly from re-estimating population statistics, with the fine-tuning (FT) model benefiting

the most. Furthermore, it is observed that pre-trained models improve only moderately compared to FT, meaning that they are less influenced by biased population statistics. Interestingly, models trained with augmentations show only slight improvement after BN re-estimation, and even exhibit a decrease in the  $CS \rightarrow Fog$  setting. The fact that they are much less affected by the biased population statistics of the BN layers reinforces the claim that they learn domain-invariant features in the early layers. This effect is likely caused by an 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 scheme. Consequently, the extracted features become invariant to these properties, causing subsequent batch normalization layers to be less affected by the distribution shift.

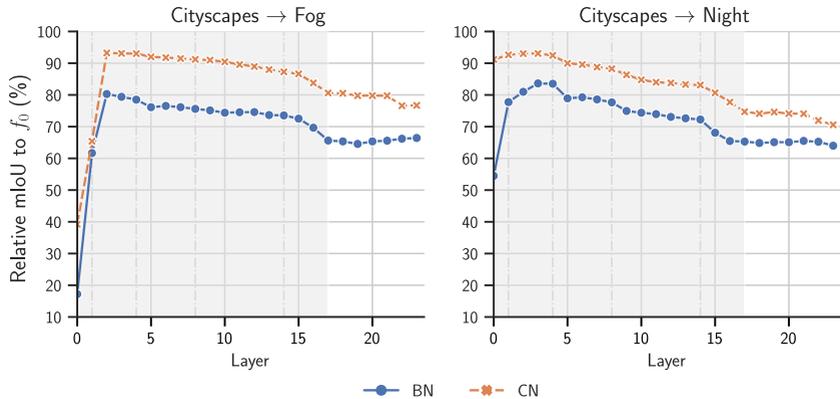
However, the fact that only re-estimating batch normalization layers without changing the initial layers leads to such a significant improvement for the remaining methods demonstrates that the adjusted population statistics can normalize the variance between the domains.

To study which batch normalization layer is most affected by the changing population statistics, the BN statistics are re-estimated for one layer at a time. Figure 6.12 reveals that the first batch normalization layer has the greatest impact on forgetting and that the last BN layer in the first block of each stage (e.g. *layer2.0.downsample.1*) has a comparable impact when the remaining batch normalization layers are not adjusted. These specific layers coincide with blocks that were identified as critical layers by Zhang et al. [Zha19b].



**Figure 6.12:** Change in mIoU on the first task after re-estimation of the population statistics of specific BN layers (horizontal axis). Re-estimation mostly affects the first BN layer and the last BN layers in each block’s first layer.

Overall, this means that batch normalization is a major contributor to forgetting in the domain-incremental setting, but forgetting is also precipitated by low-level features that are tuned to their specific domain, which lead to a major change in population statistics in the batch normalization layers. These findings are validated by exchanging all batch normalization layers with continual normalization (CN) layers [Pha22]. The layer-stitching plots in Fig. 6.13 and the results in Tab. 6.10 confirm that CN greatly reduces forgetting on CS, but the initial discrepancy in low-level features remains similar as for the models with batch normalization in the  $CS \rightarrow Fog$  setting. However, due to the combination of group and batch normalization the changing low-level features are normalized across the channel dimensions before affecting the population statistics of batch normalization.



**Figure 6.13:** Comparing the activation drift between models trained with batch normalization and continual normalization on *Night* and *Fog*. CN effectively reduces forgetting by mitigating the biased population in statistics in early BN layers.

**Table 6.10:** Results on *CS* → *ACDC* in mIoU (%) with BN and CN. By reducing the biased population statistics in early BN layers, CN effectively reduces forgetting.

Normalization	CS	Night				Fog		
	Test mIoU	Zero Shot	Test mIoU	forgetting	Zero Shot	Test mIoU	forgetting	
BN	72.0	10.5	43.6	26.1	33.4	69.0	28.0	
CN	71.2	10.3	44.2	21.2	34.4	67.1	19.1	

## 6.2.7 Layer Freezing Experiments

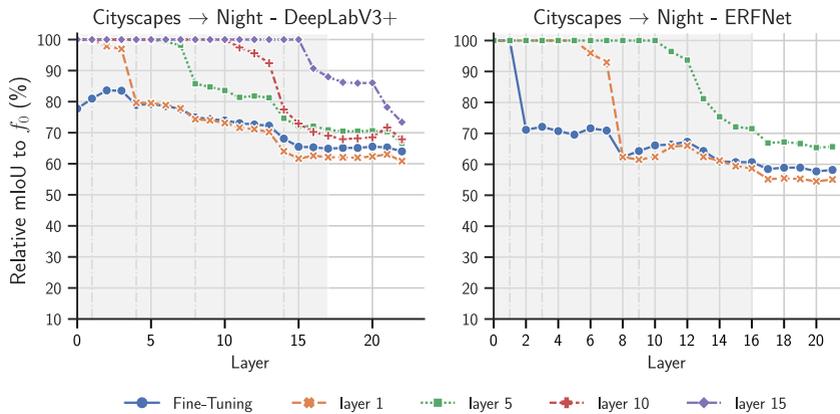
Previous experiments have shown that a major cause of forgetting is the representation shift in the early layers of the model. So the question arises: what happens if the early layers of the models are simply frozen and the population statistics are fixed during training on the ACDC subsets? Therefore, in a set of experiments, an increasing number of layers are frozen during training on the *Night* and *Rain* subsets, starting from the very first layer.

**Table 6.11:** Performance and forgetting in mIoU (%) on *CS* and the ACDC subsets *Night* and *Rain*, when the early layers of the models are frozen during fine-tuning to the new task.

Model	Frozen until	Night			Rain		
		Cityscapes mIoU	Night mIoU	Forgetting	Cityscapes mIoU	Rain mIoU	Forgetting
Deep-LabV3+	—	45.9	43.6	26.1	38.8	57.8	33.2
	1	43.6	44.5	28.4	38.5	57.5	33.5
	5	47.7	42.7	24.3	42.9	56.2	29.1
	10	48.9	39.0	23.1	50.4	52.6	21.6
	15	53.1	32.5	18.9	59.1	47.0	12.9
ERFNet	—	37.1	41.7	31.3	31.9	53.7	36.5
	1	38.8	41.3	29.6	28.6	54.6	39.8
	5	36.8	37.7	31.6	26.3	52.9	42.1
	10	44.9	36.6	23.5	49.6	48.8	18.8

The results in Tab. 6.11 show that freezing the first few layers of the encoder has only a minor effect on reducing forgetting or inhibiting learning on the new task. Only when freezing a larger number of layers in the encoder, the model is less affected by forgetting, but in turn it is also inhibited from adapting to the new task.

The reason why the effect is not as prominent for early layers can be seen in the layer stitching plots in Fig. 6.14. The representational shift of the initial layers is shifted to specific later layers, where the similarity drops down to the level of the non-frozen model. The layers where this representation shift occurs coincide with the layers that were most affected by the BN re-estimation. These results indicate that the low-level feature change cannot be addressed by freezing early layers, as it will inhibit learning the new task and shift the activation drift to later layers.



**Figure 6.14:** Activation drift between  $f_1$  to  $f_0$  measured by relative mIoU on the first task of the models stitched together at specific layers (horizontal axis). During training on *Night* the layers of ERFNet and DeepLabV3+ are frozen starting from the indicated blocks. The results demonstrate that freezing layers during training on the new task shifts the initial representation shift to later layers.

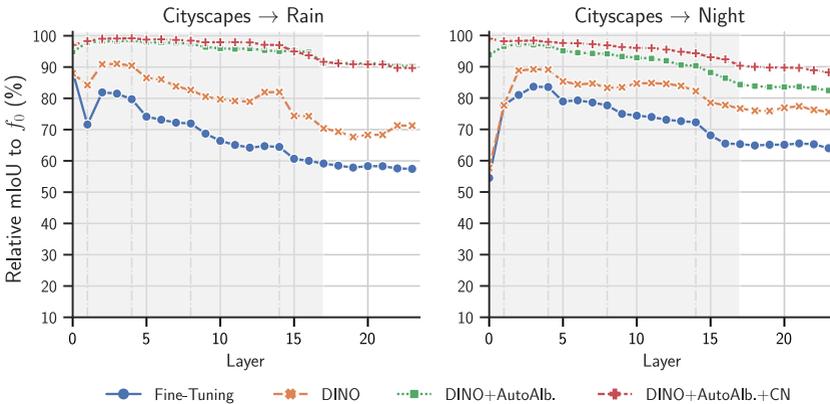
## 6.2.8 Combining the Findings

Previous experiments are repeated while making incremental changes to the training process in CS by sequentially adding pre-training with DINO, then AutoAlbum and replacing BN with CN layers. The results in Tab. 6.12 demonstrate that these changes complement each other as they dramatically reduce forgetting on CS. This is evident when comparing the layer stitching plots in Fig. 6.15, which show 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, leading to a significant reduction of activation drift throughout the entire model. This indicates that pre-training and training with augmentations enable feature reuse at different layers of the network depending on the task at hand. Specifically, augmentations seem to primarily

reduce activation drift in the first layer that is known to extract low-level features such as edges, corners and colors, whereas pre-training mitigates activation drift in intermediate layers that extract more abstract features, which potentially correspond to more complex patterns or object parts [Ola17]. Therefore, when combining pre-training, CN and augmentation, feature reuse is increased in all domains, reducing forgetting without any additional continual learning algorithm, which is demonstrated in Fig. 6.8.

**Table 6.12:** Forgetting and Learning accuracy on *CS*  $\rightarrow$  *ACDC* with incremental additions that increase the feature reuse, significantly reduces forgetting.

Method	Night		Rain		Fog		Snow	
	ln. acc.	ln. forg.						
FT	57.8	26.1	64.9	33.2	70.5	28.0	67.2	29.8
+ DINO	62.3	18.7	69.7	23.4	73.7	19.6	71.0	27.1
+ AutoAlb.	62.2	13.5	71.2	8.2	73.9	11.7	71.2	13.1
+ CN	61.3	9.1	71.0	8.0	75.4	7.3	70.9	10.5



**Figure 6.15:** Comparison of the activation drift between models trained on *Night* and *Fog* with sequentially adding pre-training with DINO, AutoAlb. and CN.

## 6.2.9 Ablation Studies

### *Comparison to Continual Learning Algorithms*

So far, no explicit continual learning strategies like regularization or replay have been used. Therefore, Tab. 6.13 compares the proposed training strategy to replay and EWC and how they perform when combined with these approaches. Interestingly, achieving low-level feature reuse outperforms regularization methods even when using the same initialization with DINO. However, the proposed training regime is outperformed by naive replay. While EWC significantly improves when combined with CN and AutoAlbum, replay does not benefit from these adjustments. For replay only pre-training leads to a significant increase in learning accuracy and a minor reduction in forgetting. A likely explanation is that the model trained with replay is able to learn domain-invariant features due to the batch construction during replay, in which half of the mini-batch consists of replay samples from previous tasks. This would also explain why replay has been shown to be sample efficient in Ch. 4.

**Table 6.13:** Forgetting and learning accuracy on *Night* and *Rain* with EWC and replay. The proposed training scheme outperforms EWC on this benchmark and can be added to existing CL methods to improve learning accuracy and reduce forgetting.

Method	Night		Rain	
	<i>lrn.</i>		<i>lrn.</i>	
	<i>acc.</i>	<i>forg.</i>	<i>acc.</i>	<i>forg.</i>
EWC	52.8	18.0	61.4	21.2
+ DINO	58.1	10.0	66.0	11.3
+ AutoAlb + CN	58.5	4.8	69.1	4.9
Replay	58.2	5.6	64.2	2.9
+ DINO	62.4	4.2	70.8	2.1
+ AutoAlb + CN	61.8	4.1	69.9	3.8
DINO + AutoAlb + CN	62.1	7.9	71.0	8.0

### *Effect of Different Architectures*

In order to verify if the obtained results can be applied to other architectures, the previous experiments are repeated with SegFormer-B2 [Xie21] and

ERFNet [Rom18], which are pre-trained on ImageNet. SegFormer-B2 was chosen as it has a similar amount of parameters as DeepLabV3+, but is a recently introduced vision transformer that utilizes self-attention layers instead of convolutional layers. The much smaller ERFNet is chosen as it is known to be more susceptible to forgetting due to its size, as it was confirmed in initial experiments in Sec. 4.2.

**Table 6.14:** Forgetting and learning accuracy of different SegFormer-B2, DeepLabV3+ and ERFNet trained on  $CS \rightarrow ACDC$  in mIoU (%) for each subset of ACDC.

Model	Night		Rain		Fog		Snow	
	<i>lrn.</i> acc.	<i>forg.</i>	<i>lrn.</i> acc.	<i>forg.</i>	<i>lrn.</i> acc.	<i>forg.</i>	<i>lrn.</i> acc.	<i>forg.</i>
SegFormer-B2 [Xie21]	59.4	16.2	69.1	11.1	71.6	11.8	70.4	14.3
DeepLabV3+ [Che18]	60.5	19.1	67.4	22.5	71.1	23.8	70.0	28.8
ERFNet [Rom18]	56.6	29.0	63.2	36.2	67.5	30.1	64.5	59.5

The results in Tab. 6.14 show that transformer-based SegFormer-B2 is much less affected by catastrophic forgetting than its CNN counterparts, even without any augmentations. A likely explanation is that the different architecture of SegFormer-B2 enables the model to learn more general features, which are more robust to distribution changes, even without any changes to the training scheme. This would also explain why SegFormer-B2 does not improve as significantly as DeepLabV3+ with the addition of pixel-level augmentations, as shown Tab. 6.15. However, it is unclear whether this is an inherent feature of the self-attention mechanism, a result of training recipes [Bai21], or a result of architectural choices such as using layer instead of batch normalization.

**Table 6.15:** Forgetting and learning accuracy in mIoU (%) of SegFormer-B2 with different augmentations. SegFormer-B2 improves less than DeepLabV3+ + using augmentations, suggesting it is less affected by color-dependent features.

Augment.	Night		Rain		Fog		Snow	
	<i>lrn.</i> acc.	<i>forg.</i>	<i>lrn.</i> acc.	<i>forg.</i>	<i>lrn.</i> acc.	<i>forg.</i>	<i>lrn.</i> acc.	<i>forg.</i>
—	59.4	16.2	69.1	11.1	71.6	11.8	70.4	14.3
Distort.	59.6	14.0	68.8	9.6	72.6	10.1	70.3	10.9
AutoAlb.	59.7	14.2	68.3	8.5	72.2	7.9	70.8	10.5

The experiments on the effect of pre-training and augmentation are also repeated on several different CNN architectures, namely ERFNet [Rom18], BiSeNet V2 [Yu21] and HRNetV2-W48 [Wan19]. The results are displayed in Tabs. 6.16 to 6.18. Similar to before, the networks are selected as they have very distinct architectures compared to DeepLabV3+, HRNetV2-W48 and BiSeNet V2 use multiple parallel branches, ERFNet has a significantly lower number of parameters.

Overall, the results confirm observations made for DeepLabV3+, by which augmentations and pre-training significantly reduce forgetting also for those selected architectures. Specifically, the combination of pre-training and AutoAlbum leads to significant improvements for all models across all datasets. However, ERFNet and BiSeNet V2 are much more affected by catastrophic forgetting, most likely due to their much smaller size.

Besides this difference, the general observations remain similar, as Distortion and AutoAlbum are the most effective methods to enforce effective feature reuse and thus a reduction of forgetting. Moreover, ImageNet pre-training again leads to higher mIoU on the target dataset but is not as effective at reducing forgetting compared to the models trained with augmentation. The only noticeable difference between the results of BiSeNet V2, ERFNet and DeepLabV3+ is the worse performance on *Snow*, which is drastically worse than the performance of the different subsets, although the same training regime is used. These results, combined with the observation that SegFormer-B2 is less affected by the domain shift, demonstrate that while the results are applicable to different CNN architectures using batch normalization, the severity of catastrophic forgetting significantly varies between architectures, as previous work has indicated [Mir22b]. Chapter 7 analyzes these findings in more depth.

**Table 6.16:** Results of ERFNet [Rom18] on  $CS \rightarrow ACDC$  in mIoU (%) for each subset of ACDC using different pre-training and augmentations strategies (Augment.). Compared to DeepLabV3+, ERFNet is much more affected by forgetting, specifically on *Snow*. However, augmentations and pre-training show the same effects as for the previous experiments.

Method	ERFNet												
	Cityscapes	Night			Rain			Fog			Snow		
	Test mIoU	Zero Shot	Test mIoU	forg.									
FT	68.4	8.2	41.7	31.3	19.5	53.7	36.5	15.2	58.0	35.3	9.8	57.1	57.4
AutoAlb.	64.0	14.4	42.6	18.9	30.5	54.4	14.7	32.9	56.4	16.4	22.7	55.7	25.1
Distort	65.7	17.7	42.7	19.3	31.0	52.5	18.0	34.9	58.5	19.4	25.3	55.7	22.6
Gaus	65.0	6.1	40.4	27.3	17.3	54.2	41.4	14.1	57.8	28.4	8.1	56.0	43.2
Noise	65.4	3.6	42.7	27.8	20.8	51.8	37.7	18.6	55.6	32.9	15.6	56.4	49.8
ImageNet	70.4	10.7	42.8	29.0	25.7	56.1	36.2	26.1	64.6	30.1	17.8	58.6	59.5
MOCO	71.8	10.2	43.0	28.4	21.7	55.8	34.9	21.3	61.7	30.4	14.0	60.4	38.4
DINO	70.1	7.6	43.3	26.3	24.3	56.6	45.8	20.8	58.9	30.7	15.6	59.6	46.9
CN	70.4	9.6	40.4	21.7	27.5	52.7	15.4	27.8	61.9	17.9	12.2	59.5	20.8
Combined	69.8	11.6	43.2	15.0	37.6	57.5	8.0	44.3	65.5	11.3	32.7	59.8	17.2
Replay	68.4	8.2	39.3	8.8	19.5	53.9	7.7	15.2	58.7	8.0	9.8	58.1	7.2
Offline		40.1	43.1	15.6	50.5	55.1	19.9	58.1	61.5	14.9	53.6	55.8	23.3

**Table 6.17:** Results of BiSeNet V2 [Yu21] on  $CS \rightarrow ACDC$  in mIoU (%) for each subset of ACDC using different pre-training and augmentations strategies. Compared to DeepLabV3+, BiSeNet V2 is more affected by forgetting.

Method	BiSeNet V2												
	Cityscapes	Night			Rain			Fog			Snow		
	Test mIoU	Zero Shot	Test mIoU	forg.									
FT	67.5	4.9	41.2	33.7	18.8	52.1	40.7	14.7	57.3	39.4	9.3	58.1	58.9
AutoAlb.	66.6	12.8	41.0	26.2	35.5	53.5	23.5	39.3	60.2	33.8	27.1	56.6	46.1
Distort	68.2	14.8	42.4	29.7	32.9	52.9	35.3	38.0	58.1	29.2	23.0	58.3	35.8
Gaus	67.1	3.8	40.8	34.2	17.6	52.9	41.2	13.9	59.4	48.1	11.0	59.1	58.9
ImageNet	69.5	7.0	42.1	35.7	20.2	54.7	49.9	14.0	60.8	46.3	13.7	57.9	62.8
CN	68.7	5.4	37.0	26.9	30.4	51.5	18.0	25.5	54.8	23.1	18.7	54.4	25.4
Combined	68.0	13.6	38.6	21.7	36.7	53.2	13.7	44.0	58.8	17.5	29.6	53.2	22.2
Replay	67.5	4.9	40.0	10.7	18.8	51.6	6.2	14.7	50.7	8.3	9.3	58.5	8.3
Offline		39.7	43.8	17.4	52.3	52.4	13.0	59.8	62.9	21.1	56.8	60.3	56.8

**Table 6.18:** Results of HRNetV2-W48 [Wan19] on  $CS \rightarrow ACDC$  in mIoU (%) for each subset of ACDC using different pre-training and augmentations strategies. HRNetV2-W48 performs similar to DeepLabV3+ on Cityscapes, but overall is more impacted by forgetting. The combination of ImageNet pre-training, AutoAlbun and continual normalization (*Combined*) leads to a significant reduction of forgetting.

Method	HRNetV2-W48												
	Cityscapes	Night			Rain			Fog			Snow		
	Test mIoU	Zero Shot	Test mIoU	forg.									
FT	70.7	6.1	42.1	38.0	22.8	59.7	37.2	19.9	67.0	36.2	15.7	62.7	44.1
AutoAlb.	72.4	19.6	44.8	33.1	43.0	58.1	12.6	55.4	68.2	15.2	37.5	61.8	20.4
Distort	70.4	15.7	44.8	21.7	33.3	58.9	13.0	38.6	64.3	11.7	24.5	62.9	18.1
Gaus	69.4	7.8	45.1	28.8	24.3	59.6	32.4	24.5	66.9	26.6	15.7	61.6	40.8
ImageNet	71.1	6.9	46.2	26.2	26.0	58.6	31.9	26.0	66.2	25.8	19.8	60.4	51.0
CN	70.5	9.8	41.9	17.0	29.9	57.0	13.1	28.1	65.9	17.1	19.7	58.0	21.8
Combined	71.8	17.7	41.9	10.4	46.3	60.4	9.3	56.9	66.6	11.1	41.3	62.1	11.3
Replay	70.7	6.1	45.2	9.8	22.8	59.2	3.3	19.9	68.9	4.4	15.7	63.3	5.9
Offline		44.8	45.6	32.5	57.9	57.9	2.4	62.0	68.8	2.4	58.2	63.1	4.3

### Longer Task Sequences

Until this point, the effect of feature reuse was only demonstrated on a training sequence of two tasks. To determine if the effects still persist in longer task sequences, the training schemes are also evaluated on a multi-step domain increment with *CS*, *Rain* and *Night*, where augmentations are only used during training on *CS*. Table 6.19 shows that pre-training and augmentation can decrease forgetting in longer task sequences, reducing forgetting not only for the initial task, but for the intermediate task as well. This indicates that once general low-level features are learned, their benefits remain even after the model is fine-tuned on a new domain without the additional augmentations. However, it should be noted that the interaction between these domains can be intricate, as a reduction in forgetting on *CS* is observed after the model was trained on *Night* when no augmentations are used. Furthermore, Nguyen et al. [Ngu19] have already identified that the order or similarity of the tasks in continual learning further impact the severity of forgetting.

**Table 6.19:** Results for CS  $\rightarrow$  Rain  $\rightarrow$  Night with DeepLabV3+. Pre-training with DINO, AutoAlbum and continual normalization (denoted as *Combined*) drastically decreases forgetting even in longer task sequences.

Method	CS	Rain		Night		
	Test mIoU	CS forg.	Test mIoU	CS forg.	Rain forg.	Test mIoU
FT	72.0	33.2	57.7	27.8	24.9	45.3
AutoAlb.	72.2	10.7	59.4	15.2	18.2	47.4
Distort	71.7	19.0	60.9	20.8	26.6	47.5
ImageNet	73.9	22.5	60.9	26.1	23.4	46.1
MOCO	75.2	26.8	63.5	18.2	20.1	47.2
DINO	75.0	23.4	64.4	18.3	21.1	49.7
CN	71.2	12.7	58.6	21.1	25.9	43.4
Combined	73.7	6.4	67.8	9.4	16.7	49.8

## 6.2.10 Conclusion

This section demonstrated that the major cause of catastrophic forgetting in domain-incremental learning is a severe drift of representations in the early layers that is affecting the first convolution layer and the population statistics of subsequent BN layers. This feature drift is precipitated by the change in low-level image statistics between the domains, which the model adapts to when trained on the new domain.

To address this problem, various pre-training schemes and pixel-level augmentations were utilized to facilitate features in early layers that can be reused in upcoming tasks. The experiments in this section showed that these methods were effective in reducing representation shift, with augmentations stabilizing the first layers and pre-training primarily stabilizing the representations after the first BN layer.

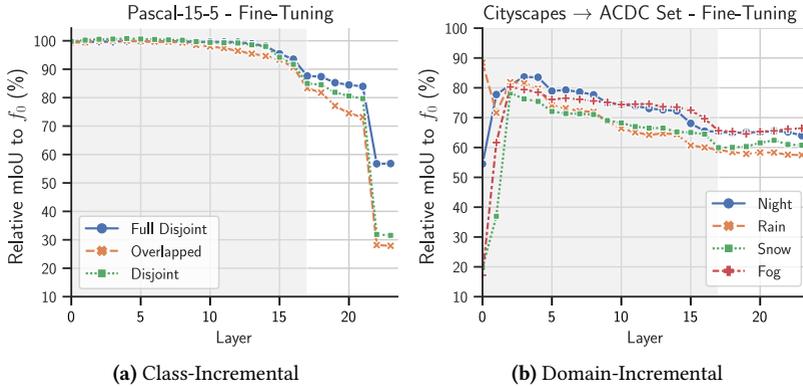
The findings suggest that training with augmentation strategies like *Distortion* or *AutoAlbum* encourages the model to learn features that are invariant to low-level image statistics such as hue, saturation and brightness that vary between the domains. Thereby, during optimization on the new domain, those features are not affected, leading to a significant reduction in forgetting. Interestingly, it was 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. This suggests that pre-training on ImageNet leads to more generalized features throughout the network. In the 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 [Gal21], but also for continual semantic segmentation.

Therefore, as pre-training and augmentations enforce feature reuse in different layers of the network, combining them leads to a significant reduction in catastrophic forgetting. In the experiments, the improved training scheme led to an average reduction of forgetting of around 20 % mIoU across the ACDC domains compared to simply fine-tuning. Overall, these 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. These generalized features will ease adaptation to new tasks while at the same time reducing the effects of catastrophic forgetting. The importance of good initialization can also be linked to the observed *critical learning periods* of deep neural networks, during which a disturbance of low-level image statistics such as blurring leads to an irreversible deficiency in the performance of the trained network [Kle23, Ach19].

### 6.3 Differences in Domain- and Class-Incremental Learning

Figure 6.16 shows a comparison of the activation drift for class- and domain-incremental learning. It is apparent that the activation drift in the domain-incremental setting mostly affects the first layers of the network, while in class-incremental learning later layers are affected. However, in these experiments the task increments were chosen so that the model increments were either pure domain or class increments. In reality, those task increments will entail a mix of changes in the input and output distributions at the same time.



**Figure 6.16:** Comparison of the activation drift a model is subjected to when it is simply fine-tuned in (a) class-incremental and (b) domain-incremental learning.

Therefore, it can be assumed that the activation drift in mixed settings will likely affect layers near the input as well as layers close to the output of the neural network, so that potential continual learning algorithms should not focus on either class- or domain-incremental settings but should provide a more holistic approach that is able to address distribution changes in the input and output space, as for example class-incremental approaches that are based on knowledge-distillation on the output layers cannot account for the severe activation drift that can arise in some domain-incremental settings.

Furthermore, Sec. 6.2 found that an important ingredient to improve the performance in continual learning in both stability and plasticity is a good initialization for the first task that enables to learn more general features that enable feature reuse. While Sec. 6.1 discovered in the class-incremental setting that without any form of replay the model is not able to effectively distinguish between all classes as it suffers from inter-task confusion, a similar limitation could not be found in domain-incremental learning.

Finally, in both settings the choice of architecture also affected the severity of catastrophic forgetting as well as how and where the activation drift manifests itself in the layers of the neural network. Especially the vision

transformer-based methods seemed to be more robust against catastrophic forgetting. Therefore, the next chapter will further analyze the effect that the architecture choice has on the performance in continual learning.



## 7 Effects of Architecture in Continual Learning

The vast majority of ongoing research in continual semantic segmentation is focused on developing new learning algorithms to mitigate the effects of catastrophic forgetting, while using largely similar CNN architectures. In contrast, previous chapters have indicated that the effects of catastrophic forgetting vary notably between different architectures. The results in Ch. 4 indicate that models with a higher number of parameters are more robust towards catastrophic forgetting; Sec. 6.1 demonstrated that inputs from multiple encoder stages increase the activation drift in early layers of the model, which precipitates catastrophic forgetting. Finally, the ablation studies in Sec. 6.2 suggest that recent VTs are potentially more robust to catastrophic forgetting, as they learn more general features without any additional augmentations and that normalization layers can have adverse effects on the performance in continual learning.

Therefore, this chapter studies how the recent developments in computer vision models such as VTs, modern CNNs, hybrid models, new normalization layers and different decoder architectures affect catastrophic forgetting in continual semantic segmentation and how the architecture influences the effects discovered in previous chapters.

## 7.1 Related Works

### 7.1.1 Differences between CNNs and Vision Transformers

With the well-established CNN and the rise of the VT, there are currently two competing architectures in the field of computer vision, each with very distinct properties. Prior and ongoing research extensively studies how the performance of these architectures differs on various computer vision tasks. Recent work claims that VTs are more robust than CNNs, specifically to severe domain shifts [Zha22, Li22a], adversarial attacks [Nas21, Pau22] or perturbations like blurring or noise [Bho21, Xie21]. At the same time Bai et al. [Bai21] have shown that results in previous work has been distorted because of differently used training regimes in prior comparisons and claim that when similar training regimes are used that CNNs can match the adversarial robustness of VTs, but that VTs achieve better generalization capabilities when pre-trained on large datasets. Wang et al. [Wan22a] identify that the improved generalization capability of VTs is not solely a property of the self-attention layer, but is to a large degree caused by other architecture differences between CNNs and VTs.

Other related works discuss the distinct properties of VTs and CNNs. It was discovered that CNNs have a texture bias, while VTs have a shape bias [Gei19, Nas21] and that self-attention layers in VTs aggregate local information in early layers similar to convolutions in CNNs, but also utilize global information much earlier than their CNN counterparts [Rag21].

### 7.1.2 Architectures in Continual Learning

Previous work on architectures for continual learning optimized the model design using neural architecture search to efficiently share or expand the model for continual learning [Xu18, Mun21].

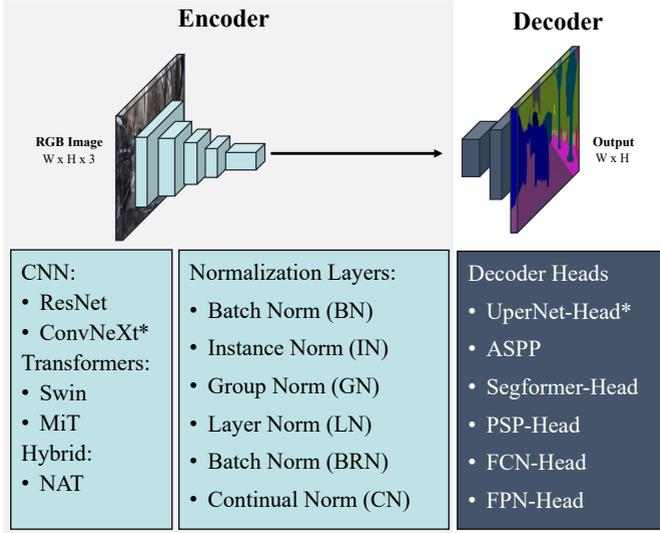
Mirzadeh et al. [Mir22b] investigate the role of architectures in continual learning, but they only investigate at nominal recognition tasks rather than

semantic segmentation tasks. Furthermore, they conduct their study on simply stacked convolutional layers rather than a backbone architecture. While they include ViT [Dos20] in their study, they do not consider other VTs or hybrid architectures. A report from the ICCV 2021 Challenge SSLAD-Track3B suggests that the recent Swin Transformer suffers less from forgetting than its CNN counterparts and thus performs better in continual learning [Li22b].

## 7.2 Experiments

This chapter examines how different neural network architectures and normalization layers perform in continual semantic segmentation. The experiments are conducted in a class- and a domain-incremental setting, as previous chapters have shown that catastrophic forgetting affects neural networks differently in these scenarios.

## 7.2.1 Experimental Setup



**Figure 7.1:** Experimental setup and architecture choices to measure the effect of architecture on catastrophic forgetting. The default choices are marked with \*. In the experiments on the encoder architecture, the same UPerNet decoder head is used and only the backbone is exchanged with different VT, CNN and hybrid architectures. The impact of the decoder is evaluated on top of a ConvNeXt backbone.

### Architectures

To evaluate the influence of different architecture design choices in continual learning, the following experiments separately evaluate the impact of different encoders, decoders and normalization layers. The selection of encoders, decoders and normalization layers is illustrated in Fig. 7.1.

In the experiments that evaluate the effect of different backbone architectures for continual semantic segmentation, only the encoder is exchanged while the

same UPerNet [Xia18] decoder head is used. For the encoder different backbones are selected from various types of CNNs, VT and hybrid architectures to compare the effect of the inherent architectural differences for continual learning.

The following architectures are chosen. As representative for CNNs:

- ResNet-50 [He16]: the most widely used CNN backbone architecture that introduced the residual connections that enabled deeper neural networks. It was until recently the widely used backbone for continual semantic segmentation.
- ConvNeXt-T [Liu22]: an modernized version of ResNet that improves the ResNet architecture with design decisions from VTs.

As representative for the VTs:

- Swin Transformer [Liu21]: a computationally more efficient version of ViT [Dos20] that was identified to be less affected by catastrophic forgetting than the CNN counterparts in a recent continual learning challenge [Li22b].
- MiT-B2 [Xie21]: the backbone architecture that was introduced with the SegFormer – which contrary to the other backbones – was specifically designed for segmentation tasks.

Finally, for the hybrid architectures:

- NAT [Has23]: the neighborhood attention transformer (NAT) that introduces inductive biases similar to CNN using overlapping windows and applies attention within the window using a novel attention mechanism called neighborhood attention.

The backbone architectures are chosen to have a similar number of parameters to ensure a fair comparison between them. The selected architectures are summarized in Tab. 7.1 along with the type of architecture and number of parameters in the network.

**Table 7.1:** Overview of the selected encoder architectures with their respective top-1 ImageNet accuracy in (%)

Type	Name	#params	FLOPs	Top-1
CNN	ResNet-50	23M	4.1G	78.8
	ConvNeXt-T	29M	4.5G	82.1
Transformer	Swin-T	28M	4.5G	81.3
	MiT-B2	25.2M	4.0G	81.6
Hybrid	NAT-T	30M	4.3G	83.2

Sec. 6.1 has shown that in the class-incremental setting, forgetting occurs primarily in the decoder stages of a model. Therefore, the effect of different decoder architectures is investigated separately, without changing the ConvNeXt-T backbone. Specifically, UPerNet, FPN and SegFormer-Head are chosen as decoders that receive input from multiple stages of the encoder and ASPP, FCN and PSPNet are chosen as decoders that only receive input from the last stage of the encoder. An overview of the different decoders is displayed in Tab. 7.2. It should be noted that contrary to the chosen encoders, the decoders vary greatly in their number of parameters.

**Table 7.2:** Overview of the selected decoder architectures with the number of parameters.

Decoder	Multi-Stage Input	#params
UperNet-Head	yes	31.4M
FPN-Head	yes	1.4M
SegFormer-Head	yes	1.8M
FCN-Head	no	11.8M
PSPNet-Head	no	14.6M
ASPP	no	23.2M

Finally, the choice of the normalization layers is evaluated, as the previous chapter has highlighted that batch normalization layers also significantly contribute to catastrophic forgetting. In this set of experiments, the batch normalization layers of ResNet-50 are replaced with *continual* [Pha22], *group* [Wu18b], *instance* [Uly16], *layer* normalization or *batch* renormalization layers [Iof17] that were introduced in Sec. 2.2.1.

## Datasets

Similarly, to previous chapters, the experiments are conducted in a class- and domain-incremental setting. In the class-incremental setting, the models are trained on the *overlapped* Pascal-15-5 split that was introduced in Sec. 6.2.

In the domain-incremental experiments, the models are adapted from good to adverse weather conditions using the incremental Cityscapes (CS) [Cor16] to ACDC [Sak21] setup. Contrary to the setup in Sec. 6.2, the subsets *Night*, *Rain*, *Snow* and *Fog* are combined into one task.

## Optimization Strategy

To achieve a fair comparison, the models are all trained using the AdamW [Los19] optimizer with a batch size of 6 and a base learning rate of  $1 \times 10^{-4}$ , which is tuned for the different backbone architectures. During an initial warm-up phase of 1500 iterations, the learning rate is slowly increased to the target learning rate. After the warm-up phase, a linearly decaying learning rate schedule is utilized. The models are trained for 120 epochs in the domain-incremental setting and 40 epochs in the class-incremental setting. The images are scaled and cropped to the size of  $769 \times 769$  for the domain-incremental setting and  $512 \times 512$  for the class-incremental setting. During training, the images are randomly flipped and scaled and *Distortion* is utilized.

The models are evaluated on the individual validation sets without using any scaling or cropping. As Sec. 6.1 has demonstrated that in the class-incremental learning setting fine-tuning the model to new classes leads to a severe background bias, the models are trained with the unbiased cross-entropy loss (UNCE) [Cer20] in the class-incremental setting.

## 7.2.2 Domain-Incremental Learning

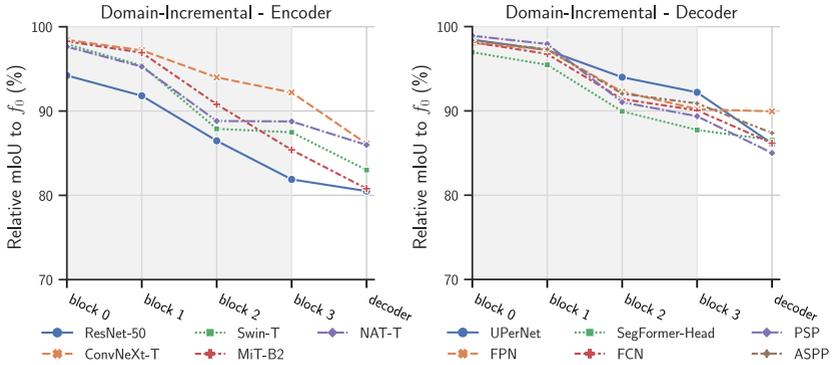
The results of the domain-incremental experiments for the different backbone architectures are displayed in Tab. 7.3, for different decoders in Tab. 7.5 and for different normalization layers in Tab. 7.4.

**Table 7.3:** Comparison of the performance of different backbone architectures in domain-incremental learning. Results are in mIoU (%).

Method	Task 1		Task 2		learning accuracy	average forgetting
	Cityscapes	ACDC	Cityscapes	ACDC		
ResNet-50	78.0 ± 0.6	36.8 ± 1.5	60.4 ± 0.3	71.9 ± 0.6	74.9	17.6
ConvNeXt-T	80.2 ± 0.6	51.3 ± 0.5	71.6 ± 0.6	72.1 ± 0.2	76.1	8.6
Swin-T	78.2 ± 0.5	44.8 ± 0.5	65.2 ± 0.6	68.8 ± 0.7	73.5	13.0
MiT-B2	78.5 ± 0.2	44.6 ± 0.9	68.1 ± 2.4	67.2 ± 0.5	72.9	10.4
NAT-T	80.2 ± 0.6	45.6 ± 1.7	68.7 ± 1.0	68.7 ± 0.9	74.4	11.5

The results for the different encoder architectures in Tab. 7.3, allow two main observations: (1) CNNs and NAT-T outperform the vision transformers in terms of *learning accuracy*, (2) ResNet (the default choice in most continual learning experiments) is most affected by catastrophic forgetting. The transformer backbones MiT-B2 and Swin-T significantly mitigate the effects of forgetting, but in turn suffer a moderate drop in *learning accuracy*. This could potentially be explained by the fact that they are known to require more data to achieve better performance than their CNN counterparts [Dos20]. The hybrid model NAT-T shows good performance for both learning accuracy and forgetting, as it re-introduces the inductive bias found in convolutional layers and therefore is not required to learn those biases in the early layers [Rag21, Has23].

Most strikingly, the CNN-based ConvNeXt-T achieves the best overall results in both highest *learning accuracy* and lowest *forgetting*, which indicates that the improved robustness of the transformer backbones towards forgetting might not be an inherent feature of the self-attention layer, but could be potentially attributed to other architectural changes.



**Figure 7.2:** Activation drift in domain-incremental learning for different encoders and decoders.

The layer stitching plot in Sec. 7.2.2 shows the activation drift at the very first layer of each block of the different encoders. As in all experiments the Distort augmentation is utilized, the high initial feature discrepancy that was observed in Sec. 6.2 is not visible. However, it is notable that ResNet-50 is most affected by the activation drift in the domain-incremental setting. Even the very first block of ResNet-50 suffers from a notably increased activation drift compared to the remaining backbones. The remaining encoders are less affected by activation drift, which indicates that they learned more general features in the first task that are reused for the second task. However, as this is not only true for the VTs, but also for the CNN-based ConvNeXt-T and NAT-T, this is most likely not only due to the self-attention layers. The cause for the improved generalization capability will be further discussed in Sec. 7.3.

**Table 7.4:** Comparison of the performance of different decoder architectures in domain-incremental learning. Results are in mIoU (%).

Method	Task 1		Task 2		learning accuracy	average forgetting
	Cityscapes	ACDC	Cityscapes	ACDC		
Batch Norm. (BN)	78.0 ± 0.6	36.8 ± 1.5	60.4 ± 0.3	71.9 ± 0.6	74.9	17.6
Continual Norm (CN)	77.9 ± 1.0	47.3 ± 0.6	65.2 ± 1.0	72.1 ± 1.2	75.0	12.7
Batch Remorm. (BRN)	76.6 ± 1.2	39.4 ± 1.1	59.5 ± 1.8	70.0 ± 2.1	73.3	17.1
Group Norm. (GN)	71.9 ± 0.6	38.5 ± 1.0	58.4 ± 1.1	66.6 ± 0.8	69.2	13.5
Instance Norm. (IN)	74.2 ± 0.2	47.5 ± 0.9	61.1 ± 0.1	69.7 ± 0.6	71.9	13.1
Layer Norm. (LN)	71.4 ± 0.8	34.5 ± 0.6	55.4 ± 0.8	65.4 ± 0.8	68.4	16.0

Next, the impact of the different normalization layers is evaluated for domain-incremental learning in Tab. 7.4. The results show that the batch normalization layer of ResNet-50 is most affected by forgetting while also being the best at adapting to the new domains. As demonstrated in the previous chapter, this is likely caused by a severe bias of the changing population mean and variance of the batch normalization layer for the most recent task. Batch re-normalization alleviates the discrepancy between the changing population statistics, but still reduces forgetting only very slightly. Channel normalization layers such as group and instance normalization are most effective in mitigating forgetting in the domain-incremental setting, but at the same time lead to reduced learning accuracy. Layer normalization, which is also used in the selected transformer backbones, also improves performance on previous tasks, but degrades performance on new tasks. As a whole, CN is the most effective since it matches the learning accuracy of BN while also significantly reducing forgetting. This is because CN combines the benefits of batch- and group normalization by first normalizing along the channel dimension and then normalizing along the mini-batch dimension, achieving both flexibility and stability.

**Table 7.5:** Comparison of the performance of different decoder architectures in domain-incremental learning. Results are in mIoU (%).

Method	Task 1		Task 2		learning accuracy	average forgetting
	Cityscapes	ACDC	Cityscapes	ACDC		
UPerNet-Head	80.9 ± 0.6	51.6 ± 0.5	71.9 ± 0.6	72.1 ± 0.2	76.1	9.0
FPN-Head	78.6 ± 0.2	47.0 ± 0.6	69.9 ± 0.3	66.3 ± 0.3	72.5	8.8
SegFormer-Head	78.7 ± 0.2	48.1 ± 0.3	69.5 ± 0.6	69.8 ± 0.4	74.2	9.2
FCN-Head	74.6 ± 0.2	44.7 ± 0.5	66.2 ± 0.6	62.0 ± 0.5	68.3	8.4
PSPNet-Head	73.8 ± 0.6	44.5 ± 0.8	66.1 ± 0.4	62.5 ± 0.1	68.2	7.7
ASPP	74.4 ± 0.3	45.5 ± 0.4	67.0 ± 0.6	63.2 ± 0.1	68.8	7.4

Finally, Tab. 7.5 lists the results for various decoder architectures on top of the ConvNeXt backbone. Overall, it is apparent that the decoder has only a negligible impact on *forgetting* compared to the encoder in the domain-incremental setting. However, it should be noted that the decoders that receive input from multiple stages of the encoder, such as UPerNet, FPN, and the SegFormer-Head, achieve a much higher *learning accuracy*.

### 7.2.3 Class-Incremental Learning

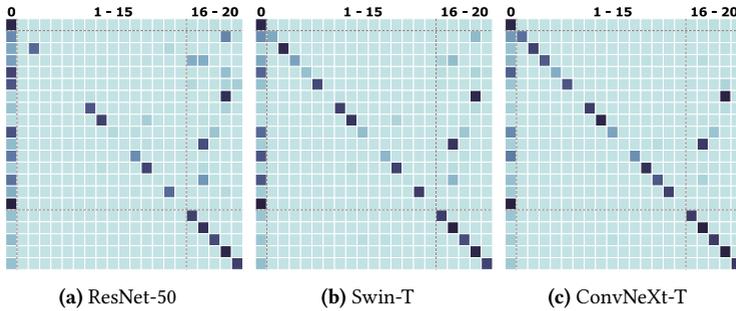
Similar as for the domain-incremental setting the results for the class-incremental setting for different encoders are displayed in Tab. 7.6, for the different normalization layers in Tab. 7.7 and for the decoders in Tab. 7.8.

**Table 7.6:** Comparison of the performance of different backbone architectures in class-incremental learning. Results are in mIoU (%).

Method	Task 1	Task 2		learning accuracy	average forgetting	
	0-15	0-15	16-20			all
ResNet-50	75.6 ± 0.8	24.7 ± 1.3	32.4 ± 1.1	26.5 ± 0.8	54.0	50.9
ConvNeXt-T	80.7 ± 0.4	47.6 ± 2.6	44.1 ± 0.9	46.8 ± 2.1	62.4	33.1
Swin-T	78.4 ± 0.3	38.6 ± 1.8	37.4 ± 1.0	38.3 ± 1.2	57.9	40.4
MiT-b2	81.2 ± 0.5	25.7 ± 6.7	39.1 ± 1.7	28.9 ± 5.0	60.2	62.5
NAT-T	80.4 ± 0.4	34.9 ± 1.9	39.2 ± 1.2	35.9 ± 1.2	59.8	42.9

As already demonstrated in previous chapters, it can be observed in Tab. 7.6 that forgetting in the class-incremental setting is much more severe than

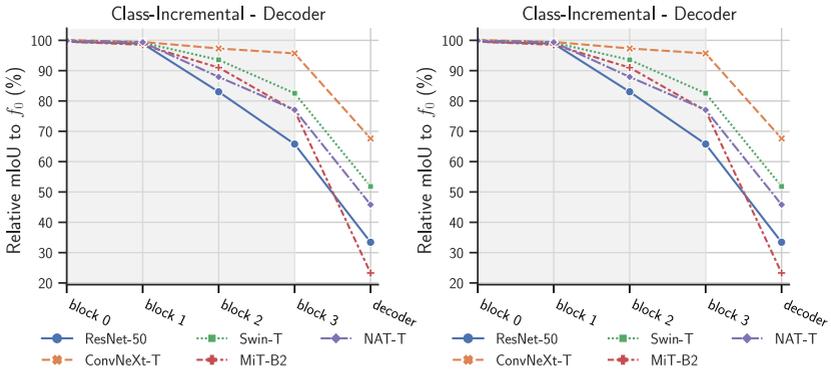
in the domain-incremental setting. Similar to the results in the domain-incremental setting, Swin-T and NAT-T, are more effective at reducing forgetting than ResNet-50. MiT-B2 is surprisingly most affected by forgetting and achieves highly deviating results on the prior classes. This is potentially caused by the small number of parameters compared to the remaining decoders. ConvNeXt performs the best in the class-incremental setting, with significant improvements compared to ResNet-50 in both learning new classes and mitigating the forgetting of old classes. Contrary to the domain-incremental results, it is observed that VTs have a higher learning accuracy than ResNet in the class-incremental setting.



**Figure 7.3:** Confusion matrices after training on PascalVoc-15-5. The confusion matrix for ResNet-50 shows a severe bias to the background class and classes of the recent task (16-20). Using Swin-T or ConvNeXt as backbone with the same decoder head decreases the bias for the new classes and the background class.

The confusion matrices in Sec. 7.2.3 demonstrate that even when using the UNCE loss, there is still a severe bias for the background class and new classes, especially for ResNet-50. However, ConvNeXt and the transformer-based models significantly reduce this bias, which indicates that the features of the VTs and ConvNeXt are more useful for the decoder to discriminate between the different classes. Furthermore, it should be noted that the only classes that ConvNeXt struggles to learn correctly are *cow* and *bus*, which are mistaken

for the new classes *sheep* and *train*, respectively. This effect, known as inter-task confusion, was already described in Sec. 6.1 and is likely a limitation in class-incremental learning without any form of replay.



**Figure 7.4:** Activation drift in domain-incremental learning for different encoders and decoders.

The layer stitching plot in Sec. 7.2.3 confirms the previous finding that the highest activation drift in class-incremental learning is observed in the later decoder layers, regardless of the chosen backbone architecture. However, ConvNeXt still notably reduces the activation drift in the encoder layers of the network, which in turn reduces catastrophic forgetting.

When comparing the different normalization layers in the class-incremental learning in Tab. 7.7, we observe that batch normalization, batch renormalization and continual normalization have higher learning compared to channel normalization layers, which again confirms that normalizing along the mini-batch dimension improves the forward transfer capability. However, unlike in the domain-incremental setting, continual normalization does not mitigate *forgetting* compared to the batch normalization or methods that normalize across the channel dimension. This is likely caused by the fact that in the class-incremental setting, the population statistics of the encoder are not

changing as much, as the input data distribution is much more similar than in the domain-incremental setting.

**Table 7.7:** Comparison of the performance of different normalization layers in class-incremental learning. Results are in mIoU (%).

Method	Task 1	Task 2			learning accuracy	average forgetting
	0-15	0-15	16-20	all		
Batch Norm. (BN)	75.6 ± 0.8	24.7 ± 1.3	32.4 ± 1.1	26.5 ± 0.8	54.0	50.9
Continual Norm (CN)	73.2 ± 0.5	20.9 ± 2.0	28.4 ± 1.6	22.7 ± 1.1	50.8	52.3
Batch Re-Norm. (BRN)	74.2 ± 0.6	19.2 ± 0.5	30.9 ± 2.0	21.9 ± 0.1	52.5	55.0
Group Norm. (GN)	61.6 ± 0.6	13.8 ± 2.2	23.6 ± 2.2	14.9 ± 1.1	42.6	47.8
Instance Norm. (IN)	65.5 ± 1.0	19.2 ± 0.6	23.6 ± 3.2	23.5 ± 4.4	44.5	46.4
Layer Norm. (LN)	65.5 ± 1.6	15.6 ± 0.7	27.2 ± 1.1	18.4 ± 0.8	45.8	48.8

Previous results indicated that in the class-incremental setting, forgetting occurs primarily in the decoder stages of a model. Therefore, Tab. 7.8 displays the performance of various decoder architectures in the class-incremental setting when using the same ConvNeXt-T backbone. Most strikingly, it is notable that the decoder heads using multi-stage input from the encoder, such as UPerNet, FPN and the SegFormer-Head, achieve higher learning accuracy but in turn suffer from more severe forgetting than decoders that only use the output from the final encoder layer. Previous work noted that representations of layers close to the output are primarily affected by a change of the training objective [Kor21]. Therefore, it can be assumed that the use of multi-stage input also leads to more severe changes in earlier stages of the encoder, as the earlier stages of the encoder are closer connected to the output layer when using input from earlier stages. The increased activation drift in early layers will in turn also affect subsequent layers, which leads to increased forgetting.

**Table 7.8:** Results in mIoU (%) for the class-incremental setting with different decoder heads.

Method	Task 1		Task 2		learning accuracy	average forgetting
	0-15	0-15	16-20	all		
UPerNet	80.7 $\pm$ 0.4	47.6 $\pm$ 2.6	44.1 $\pm$ 0.9	46.8 $\pm$ 2.1	62.4	33.1
FPN	78.9 $\pm$ 0.6	36.0 $\pm$ 5.7	41.8 $\pm$ 1.4	37.4 $\pm$ 1.4	60.4	43.0
SegFormer	78.6 $\pm$ 0.1	48.6 $\pm$ 2.6	40.9 $\pm$ 1.6	46.8 $\pm$ 2.3	59.8	30.1
FCN	74.6 $\pm$ 0.3	43.7 $\pm$ 1.2	41.8 $\pm$ 0.9	43.3 $\pm$ 0.8	58.2	30.9
PSPNet	75.5 $\pm$ 0.3	50.8 $\pm$ 2.2	41.1 $\pm$ 1.3	48.5 $\pm$ 1.9	58.3	24.7
ASPP	75.6 $\pm$ 0.3	51.6 $\pm$ 1.2	41.7 $\pm$ 2.1	49.3 $\pm$ 1.5	58.6	24.0

The layer stitching plot in Sec. 7.2.3 on the right confirms this hypothesis for FPN and the SegFormer-Head, as increased activation drift is observed in the encoder when using these decoders. Additionally, the results indicate that decoders with a higher number of parameters prevent changes in the encoder with their higher learning capacity, so that forgetting is less prominent in models with more parameters. Interestingly, models that use a moderate amount of parameters and single-stage input such as PSPNet-Head and ASPP are best at mitigating catastrophic forgetting. The results also highlight the more efficient decoder design of Segformer-Head, as it achieves a comparable performance to the UPerNet-Head while having only 1/20<sup>th</sup> of its parameters.

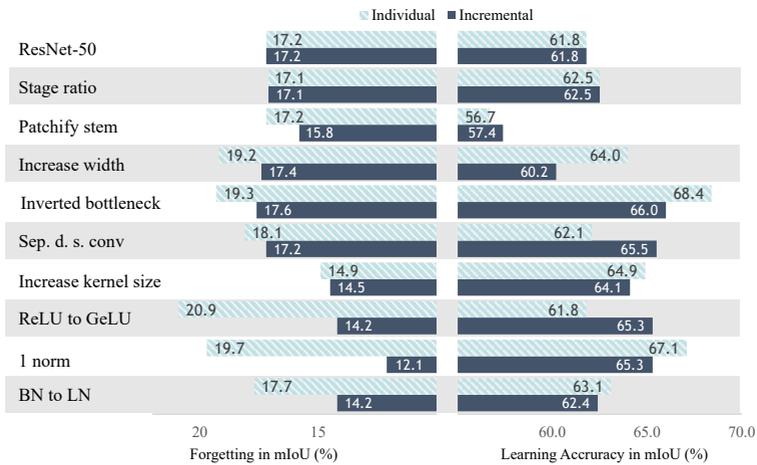
## 7.3 Ablation on ConvNeXt

The previous experiments demonstrate that the robustness towards catastrophic forgetting of SegFormer (observed in Sec. 6.2) might not be only caused by the self-attention layer, but also different architectural changes, as the performance of the CNN-based ConvNeXt clearly indicates. The following experiment aims at revealing which of the changes to ResNet-50 architecture leads to the increased performance in continual learning. Therefore, the architectural changes that were applied to convert ResNet-50 into ConvNeXt are evaluated individually and incrementally without reversing the previous changes. Table 7.9 lists the architectural changes in the order they will be applied. The resulting models are trained in the domain-incremental setting, without pre-training and starting from the same random initialization.

**Table 7.9:** Architectural changes in the ablation study ResNet-50  $\rightarrow$  ConvNeXt inspired by Liu et al. [Liu22].

No.	Name	Description
1	stage ratio	change the number of blocks in each stage from (3, 4, 6, 3) to (3, 3, 9, 3)
2	patchify stem	stem cell was replaced with a patchify layer with non-overlapping $4 \times 4$ stride 4 convolutional layers
3	increase width	Change width from 64 to 96
4	inverted bottleneck	Change the channels in the block from (64, 64, 256) to (96, 384, 96)
5	sep. d.s. conv	Use depth-wise separable convolutions to reduce computation as a trade-off for using large kernels
6	increase kernel size	Change kernel size from $[3 \times 3]$ to $[7 \times 7]$
7	ReLU to GeLU	Change the activation layer from ReLU to GeLU
8	1 norm	Reduce the number of normalization layers by removing two out of three BN layers
9	BN to LN	Change the normalization layer form batch normalization to layer normalization

The *learning accuracy* and *forgetting* for the individual and incremental changes are reported in Fig. 7.5. Some individual changes, like increasing the width from 64 to 96, inverting the bottleneck and reducing the number of normalization layers to 1 (1 norm), improve learning accuracy but also increase *forgetting*. Increasing the kernel size from  $[3 \times 3]$  to  $[7 \times 7]$  is the only individual change that increases *learning accuracy* and decreases *forgetting*. The larger kernel size in ConvNeXt attempts to replicate the global receptive field that is an inherent property of the self-attention layer of VTs, so that the network is able to capture long-range dependencies more easily. Prior work by Wang et al. [Wan22a] has demonstrated that the increased kernel size leads to increased robustness to out-of-distribution samples. Therefore, it is hypothesized that the increased kernel size leads to more generalized features in the network that are reused during incremental training, which leads to a reduction of forgetting.



**Figure 7.5:** Forgetting and learning accuracy for ResNet → ConvNeXt individual vs. incremental changes. Hatched light blue bars indicate changes that are made individually on ResNet, while solid blue bars include all previous changes. The sequence of changes is from top to bottom.

However, when the changes are made incrementally, not only does the increased kernel size lead to a reduction in *forgetting* but also to a slight improvement in learning accuracy. One of the interesting findings is that individually, the reduction of normalization layers from 3 to 1 (1 norm) leads to an increase in *forgetting* of +2.5 mIoU, but when done incrementally, it leads to an improvement in *forgetting* of -2.1 mIoU. For the incremental changes, the most effective additions were: *patchify stem*, *increasing kernel size* and *1 norm*. These changes coincide with architectural changes Wang et al. [Wan22a] proposed to make CNNs as robust to out-of-distribution samples as VTs. This further strengthens the claim that the ability of the model to mitigate forgetting depends on its ability to learn general and robust features.

Overall, this experiment concludes that the global receptive field of the VTs is a major reason for the increased robustness towards forgetting. This property can be imitated by increasing the kernel size of the convolutional layers in CNNs, which leads to similar robustness towards forgetting. Furthermore,

these experiments confirm that reducing the number of normalization layers and replacing the batch normalization layers is also vital for continual learning, which was already discussed in the previous chapter. Other changes of ConvNeXt only have minor impact on the performance in continual learning, but mainly decrease the number of computations required that are introduced with larger kernel sizes.

## 7.4 Conclusion

This chapter evaluated the effect of the choice of encoders, decoders and normalization layers on neural networks for continual semantic segmentation. It was found that the choice of architecture has a significant impact on performance in terms of learning accuracy and mitigating the effects of catastrophic forgetting. While traditional CNNs have demonstrated higher plasticity than their VT counterparts, they are more susceptible to catastrophic forgetting. However, the modernized ConvNeXt shows that this increased robustness can be replicated in CNNs by increasing the kernel size in convolutional layers, reducing the number of normalization layers, replacing batch normalization layers and by patchifying the input. These changes enable the network to learn more generalized features that, similar to those in the previous chapter, reduce catastrophic forgetting as features are reused for subsequent tasks. Even in class-incremental learning, where forgetting occurs primarily in decoder layers, the choice of the encoder architecture has a significant impact on forgetting. Additionally, the experiments demonstrate that input into the decoder from various backbone stages improves learning accuracy but increases forgetting because early layers are more susceptible to activation drift.

Overall, this chapter demonstrates the importance of architecture in the development of algorithms for continual learning and lays the groundwork for future research into architectures tailored to continual semantic segmentation.

# 8 Conclusion and Outlook

## 8.1 Conclusion

This thesis investigated the principles of catastrophic forgetting for deep neural networks in continual learning, with a focus on continual learning for scene perception in the context of automated driving. Rather than concentrating on incremental improvements to continual learning procedures, this thesis investigates concrete causes of catastrophic forgetting and their corresponding effects on neural networks, aiming to establish principles that hold true across various scenarios to further the understanding of the underlying mechanics of forgetting in deep continual learning.

A core aspect of this investigation is the development and analysis of methods that allow for quantitative measurement of activation drift and other effects of forgetting. Therefore, several existing methods and two newly proposed methods, namely layer stitching and decoder retraining, were compared in their ability to measure the effects of forgetting. It was found that layer stitching, which measures the functional similarity for each layer in a neural network, provides the most meaningful and interpretable insights, but that other methods such as BN re-estimation, decoder retraining and CKA should be used in complementary ways to infer more specific causes. These methods provide an important step towards a deeper understanding of the effects of catastrophic forgetting.

Using these methods, the main part of this thesis explored how catastrophic forgetting manifests in neural networks in class- and domain-incremental learning and how it affects their performance. In class-incremental learning, the semantic shift of the background class is identified as a major cause

of catastrophic forgetting. Layer stitching reveals that this is caused by a severe functional shift of the activations in the decoder layers, in which features that were evidence for old classes in the first task are interpreted as features for the background class after training on new classes. This effect is substantially reduced when the semantic shift of the background class is avoided in the proposed *full disjoint* setting. The experiments also indicate that continual learning methods that neither replay samples containing old classes nor utilize knowledge distillation fail to learn discriminating features for classes that are learned in different tasks and thus suffer from inter-task confusion.

In the domain-incremental setting, it is found that a major cause of forgetting is the shift of low-level representations in the first convolution layer that adversely affects the population statistics of the following batch normalization layer. To reduce this feature discrepancy in the initial layers of the model, augmentations are utilized that induce color-invariant features during training on the first task. These features are reused in subsequent tasks, which significantly reduces the activation drift in early layers and thus catastrophic forgetting. Finally, it is observed that while these augmentations stabilize features in the early layers, pre-training on ImageNet using self-supervised methods such as SwAV or DINO stabilizes intermediate representations.

Furthermore, the effect of various architecture choices is evaluated, as the previous findings already indicated that the recent VTs might be more resilient towards catastrophic forgetting. However, the experiments conclude that while VTs are more resilient than previous CNNs, such as ResNet or ERFNet, this robustness can be replicated in CNNs by increasing the kernel size of the convolution layers, reducing the normalization layers, exchanging batch normalization layers and patchifying the input. These changes precipitate more general features in the encoder so that the layers of the encoder are less affected by activation drift, which significantly reduces forgetting in class- and domain-incremental learning.

Overall, these results highlight that the principles and effects of catastrophic forgetting are nuanced as they result from a complex interplay of architectural decisions of the model, the order in which tasks are presented, and especially whether the model is adapted for new classes or new domains. The insights of

this thesis are intended to aid in the development of more robust and adaptable perception models, which will be essential for the widespread adoption of safe and reliable autonomous vehicles in the future and likely other applications of continual learning in autonomous and automated systems.

Finally, the insights of this thesis can be condensed into the following advice for the development of future continual learning algorithms:

- Continual learning algorithms should not be developed completely task-agnostic, but instead need to be tailored to the task at hand. For example in semantic segmentation it is sensible to make use of the reappearing classes in the background with knowledge distillation.
- Models used in continual learning should start from a general initialization that enables future task to reuse features from previous tasks. This can significantly avoid activation drift within the model, leading to a reduction of catastrophic forgetting.
- Continual learning algorithms should use some form of replay to avoid inter-task confusion, which is potentially a limitation of rehearsal-free continual learning.
- Random data selection for replay can be improved by balancing the classes in the buffer and choosing samples to approximate the data distribution in the embedding space of the model.
- Model architectures should be taken into consideration when designing a machine learning system for continual learning.
  - Input from multiple stages from the encoder increases the performance on the most recent task, but in turn leads to increased forgetting.
  - Architectures that achieve better out-of-distribution performance are less likely to be affected by catastrophic forgetting.
  - Batch normalization layers should be avoided and can be replaced by layer normalization or continual normalization.

## 8.2 Outlook

### *Increasing the task variety*

The experiments in this thesis are conducted on specific class- and domain-incremental benchmarks that will not conclusively prove universal properties for continual semantic segmentation. The investigations in the domain-incremental setting are limited to incremental semantic segmentation in adverse weather conditions for scene perception in automated driving. This setting introduces very distinct domain changes that were found to primarily affect the first layers of the network; it is likely that other, more abstract domain changes will affect different layers. For example, when adapting a model from German driving scenarios to Canadian driving scenarios, the class *bike lane*, changes from a red to a green color. This change will likely not require a change in early features, but adjustments in later layers in which the color green in the context of a street would need to be associated with the class *bike lane*. Therefore, future work could investigate how different domain changes affect catastrophic forgetting. Similarly, in the class-incremental setting, the experiments were conducted solely on the two-task incremental benchmark PascalVoc-15-5. These benchmarks could be expanded to different datasets with longer task sequences and a higher number of semantically diverse classes to study whether the observed effects increase in intensity. Finally, as class and domain increments will in practice often occur at the same time, future work could further investigate how this combined setting would affect deep neural networks.

### *Measure Catastrophic Forgetting During Training*

While the tools used in this thesis are capable of providing deeper insights into catastrophic forgetting, there are still some limitations in their expressiveness as well as in their application. First of all, methods like layer stitching and Dr. Frankenstein help to gain insights into the activation drift of the specific layers; however, these methods are focused on evaluating the model after it has

been trained in a continual manner. There is currently limited work on measuring catastrophic forgetting during incremental training. However, it could potentially increase the understanding of phenomena such as the *stability gap* [De 22] and expose other unexpected training dynamics that continual learning introduces. Therefore, a potential extension of this work could investigate how the activation drift materializes during the continual training process.

### ***Outlook on continual learning***

The findings and methodologies explored in this thesis are intended to pave the way for future research directions and applications in the field of continual learning. Especially at this point in time, with the emerging foundation models that potentially mark a lasting shift for the field of continual learning.

These foundation models are trained on a vast quantity of data at scale, often using self-supervised learning. They are intended to be universal feature extractors that can produce useful features for a large number of downstream tasks. They are often trained in a contrastive manner using only images or image-text pairs. A prime example is CLIP [Rad21], which is trained on 400 million image-text pairs using a contrastive learning approach that aims to predict the correct image-text pairing. The resulting model has an incredible rich feature space that can be used for zero-shot image classification, image-text retrieval and is even utilized as a prompting interface for several image generators such as Stable Diffusion [Rom22]. As models like CLIP often do not even require task-specific training data and can be utilized and adjusted via prompting, it often raises the question whether the existence of these models would make continual learning obsolete. In fact, the results of Sec. 6.2 already demonstrate that even when the models would require any fine-tuning, the fact that they already have such general features means that they are even less affected by catastrophic forgetting.

Nonetheless, this does not render continual learning obsolete, as our changing environment demands that foundation models require continued adjustment to remain relevant and accurate. However, foundation models definitely change the focus of the field. Research in continual learning will now center

on models that do not require starting from random initialization. Instead, the challenge now lies in developing novel continual learning approaches that can capitalize on these foundation models' general features while fine-tuning them for specific tasks and adapting to new information.

There are already recent works that, instead of modifying the pre-trained encoder directly, propose to utilize a small number of insert-able model instructions, called prompts [Smi23, Wan22c, Wan22b]. These prompts are generated based on the input image and then inserted into various layers of the encoder to extract task-specific features from the foundation model that are then utilized in the classifier. In this setting, the encoder remains completely fixed and only prompt keys and the classifier layers are expanded. Therefore, future work could utilize the tools proposed in this thesis to measure where these foundation models would be most affected by activation drift to indicate the layers in which these prompts should be inserted, as especially in the class-incremental setting, early layers are often not affected by activation drift.

Additionally, at the moment, the limitations of these foundation models are still unclear, so it will be furthermore important to study under which circumstances these models would be affected by catastrophic forgetting and how this would be different from the observations made in this thesis.

## Bibliography

- [12] “IBM100 - Deep Blue”. en-US. In: (Mar. 2012). Publisher: IBM Corporation. URL: %5Curl%7Bhttp://www-03.ibm.com/ibm/history/ibm100/us/en/icons/deepblue/%7D (visited on 07/16/2023) (cit. on p. 1).
- [16] “Artificial intelligence: Google’s AlphaGo beats Go master Lee Se-dol”. en-GB. In: *BBC News* (Mar. 2016). URL: %5Curl%7Bhttps://www.bbc.com/news/technology-35785875%7D (visited on 07/16/2023) (cit. on p. 1).
- [Abe21] ABELLO, Antonio A.; HIRATA, Roberto and WANG, Zhangyang: “Dissecting the High-Frequency Bias in Convolutional Neural Networks”. In: *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. 2021, pp. 863–871. DOI: 10.1109/CVPRW53098.2021.00096 (cit. on p. 109).
- [Ach19] ACHILLE, Alessandro; LAM, Michael; TEWARI, Rahul; RAVICHANDRAN, Avinash; MAJI, Subhransu; FOWLKES, Charless C; SOATTO, Stefano and PERONA, Pietro: “Task2vec: Task embedding for meta-learning”. In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2019, pp. 6430–6439 (cit. on pp. 42, 129).
- [Ach20] ACHARYA, Manoj; HAYES, Tyler L. and KANAN, Christopher: RODEO: Replay for Online Object Detection. Aug. 14, 2020. URL: <http://arxiv.org/pdf/2008.06439v1> (cit. on p. 34).
- [Alj17] ALJUNDI, Rahaf; CHAKRAVARTY, Punarjay and TUYTELAARS, Tinne: “Expert Gate: Lifelong Learning With a Network of Experts”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. July 2017 (cit. on p. 37).

- [Alj18] ALJUNDI, Rahaf; BABILONI, Francesca; ELHOSEINY, Mohamed; ROHRBACH, Marcus and TUYTELAARS, Tinne: “Memory Aware Synapses: Learning What (not) to Forget”. In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 11207 LNCS (2018), pp. 144–161. DOI: 10.1007/978-3-030-01219-9\_9. arXiv: 1711.09601 (cit. on pp. 35, 36, 48, 88, 89, 92, 96, 99, 103, 199).
- [Alj19] ALJUNDI, Rahaf; LIN, Min; GOUJAUD, Baptiste and BENGIO, Yoshua: Gradient based sample selection for online continual learning. Mar. 20, 2019. URL: <http://arxiv.org/pdf/1903.08671v5> (cit. on pp. 34, 56).
- [Ama21] AMADEO, Ron: Waymo expands to San Francisco with public self-driving test [Update]. en-us. Aug. 2021. URL: %5Curl%7Bhttps://arstechnica.com/gadgets/2021/08/waymo-expands-to-san-francisco-with-public-self-driving-test/%7D (visited on 06/03/2023) (cit. on p. 2).
- [Ba16] BA, Jimmy Lei; KIROS, Jamie Ryan and HINTON, Geoffrey E.: Layer Normalization. 2016. DOI: 10.48550/ARXIV.1607.06450. URL: <https://arxiv.org/abs/1607.06450> (cit. on p. 77).
- [Bai21] BAI, Yutong; MEI, Jieru; YUILLE, Alan L and XIE, Cihang: “Are Transformers more robust than CNNs?” In: *Advances in Neural Information Processing Systems*. Ed. by RANZATO, M.; BEYGELZIMER, A.; DAUPHIN, Y.; LIANG, P.S. and VAUGHAN, J. Wortman. Vol. 34. Curran Associates, Inc., 2021, pp. 26831–26843. URL: <https://proceedings.neurips.cc/paper/2021/file/e19347e1c3ca0c0b97de5fb3b690855a-Paper.pdf> (cit. on pp. 124, 134).
- [Ban21] BANSAL, Yamini; NAKKIRAN, Preetum and BARAK, Boaz: “Revisiting Model Stitching to Compare Neural Representations”. In: *Advances in Neural Information Processing Systems*. Ed. by BEYGELZIMER, A.; DAUPHIN, Y.; LIANG, P. and VAUGHAN, J. Wortman. 2021. URL: <https://openreview.net/forum?id=ak06J5jNR4> (cit. on pp. 67, 75).

- [Bho21] BHOJANAPALLI, Srinadh; CHAKRABARTI, Ayan; GLASNER, Daniel; LI, Daliang; UNTERTHINER, Thomas and VEIT, Andreas: “Understanding Robustness of Transformers for Image Classification”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2021, pp. 10231–10241 (cit. on p. 134).
- [Bue09] BUEHLER, Martin; IAGNEMMA, Karl; SINGH, Sanjiv; SICILIANO, Bruno; KHATIB, Oussama and GROEN, Frans, eds.: *The DARPA Urban Challenge: Autonomous Vehicles in City Traffic*. en. Vol. 56. Springer Tracts in Advanced Robotics. Berlin, Heidelberg: Springer, 2009. DOI: 10.1007/978-3-642-03991-1. URL: %5Curl%7Bhttps://link.springer.com/10.1007/978-3-642-03991-1%7D (visited on 06/03/2023) (cit. on p. 2).
- [Bus20] BUSLAEV, Alexander; IGLOVIKOV, Vladimir I.; KHVEDCHENYA, Eugene; PARINOV, Alex; DRUZHININ, Mikhail and KALININ, Alexandr A.: “Albumentations: Fast and Flexible Image Augmentations”. In: *Information* 11.2 (2020). DOI: 10.3390/info11020125. URL: <https://www.mdpi.com/2078-2489/11/2/125> (cit. on pp. 111, 201, 204).
- [Car20] CARON, Mathilde; MISRA, Ishan; MAIRAL, Julien; GOYAL, Priya; BOJANOWSKI, Piotr and JOULIN, Armand: “Unsupervised learning of visual features by contrasting cluster assignments”. In: vol. 33. 2020, pp. 9912–9924 (cit. on p. 204).
- [Car21] CARON, Mathilde; TOUVRON, Hugo; MISRA, Ishan; JÉGOU, Hervé; MAIRAL, Julien; BOJANOWSKI, Piotr and JOULIN, Armand: “Emerging Properties in Self-Supervised Vision Transformers”. In: *Proceedings of the International Conference on Computer Vision (ICCV)*. 2021 (cit. on pp. 111, 204).
- [Cas18] CASTRO, Francisco M.; MARÍN-JIMÉNEZ, Manuel J.; GUIL, Nicolás; SCHMID, Cordelia and ALAHARI, Karteek: End-to-End Incremental Learning. July 25, 2018. URL: <http://arxiv.org/pdf/1807.09536v2> (cit. on p. 34).

- [Cer20] CERPELLI, Fabio; MANCINI, Massimiliano; BULO, Samuel Rota; RICCI, Elisa and CAPUTO, Barbara: “Modeling the background for incremental learning in semantic segmentation”. In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (Feb. 2020), pp. 9230–9239. DOI: 10.1109/CVPR42600.2020.00925. arXiv: 2002.00718. URL: <https://arxiv.org/abs/2002.00718> (cit. on pp. 39, 47, 87–89, 97–99, 139, 201).
- [Cer22] CERPELLI, Fabio; FONTANEL, Dario; TAVERA, Antonio; CICCONE, Marco and CAPUTO, Barbara: “Incremental Learning in Semantic Segmentation From Image Labels”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2022, pp. 4371–4381 (cit. on p. 40).
- [Cha18a] CHAUDHRY, Arslan; DOKANIA, Puneet K.; AJANTHAN, Thalaiyasingam and TORR, Philip H. S.: “Riemannian Walk for Incremental Learning: Understanding Forgetting and Intransigence”. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. Sept. 2018 (cit. on p. 36).
- [Cha18b] CHAUDHRY, Arslan; RANZATO, Marc’Aurelio; ROHRBACH, Marcus and ELHOSEINY, Mohamed: Efficient Lifelong Learning with A-GEM. Dec. 2, 2018. URL: <http://arxiv.org/pdf/1812.00420v2> (cit. on pp. 34, 36).
- [Che17a] CHEN, Liang-Chieh; PAPANDREOU, George; KOKKINOS, Iasonas; MURPHY, Kevin and YUILLE, Alan L: “Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs”. In: *IEEE transactions on pattern analysis and machine intelligence* 40.4 (2017), pp. 834–848 (cit. on p. 27).
- [Che17b] CHEN, Liang-Chieh; PAPANDREOU, George; SCHROFF, Florian and ADAM, Hartwig: “Rethinking Atrous Convolution for Semantic Image Segmentation”. In: *CoRR* abs/1706.05587 (2017). arXiv: 1706.05587. URL: <http://arxiv.org/abs/1706.05587> (cit. on p. 48).

- [Che18] CHEN, Liang-Chieh; ZHU, Yukun; PAPANDREOU, George; SCHROFF, Florian and ADAM, Hartwig: “Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation”. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. Sept. 2018 (cit. on pp. 27, 88, 106, 124, 204).
- [Che19] CHEUNG, Brian; TEREKHOV, Alexander; CHEN, Yubei; AGRAWAL, Pulkit and OLSHAUSEN, Bruno: “Superposition of many models into one”. In: *Advances in Neural Information Processing Systems*. Ed. by WALLACH, H.; LAROCHELLE, H.; BEYGEZIMMER, A.; D’ALCHÉ-BUC, F.; FOX, E. and GARNETT, R. Vol. 32. Curran Associates, Inc., 2019. URL: <https://proceedings.neurips.cc/paper/2019/file/4c7a167bb329bd92580a99ce422d6fa6-Paper.pdf> (cit. on p. 38).
- [Che21a] CHEN\*, Xinlei; XIE\*, Saining and HE, Kaiming: “An Empirical Study of Training Self-Supervised Vision Transformers”. In: *arXiv preprint arXiv:2104.02057* (2021) (cit. on pp. 111, 204).
- [Che21b] CHENG, Bowen; SCHWING, Alex and KIRILLOV, Alexander: “Per-pixel classification is not all you need for semantic segmentation”. In: *Advances in Neural Information Processing Systems 34* (2021), pp. 17864–17875 (cit. on pp. 27, 40).
- [Che22] CHENG, Bowen; MISRA, Ishan; SCHWING, Alexander G; KIRILLOV, Alexander and GIRDHAR, Rohit: “Masked-attention mask transformer for universal image segmentation”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 1290–1299 (cit. on p. 27).
- [Che23] CHENG, Evelyn: Baidu says it can now operate robotaxis in Beijing with no human staff inside. en. Mar. 2023. URL: %5Curl%7Bhttps://www.cnbc.com/2023/03/17/baidu-says-it-can-now-operate-robotaxis-in-beijing-with-no-human-staff-inside.html%7D (visited on 06/03/2023) (cit. on p. 2).
- [Cir12] CIREGAN, Dan; MEIER, Ueli and SCHMIDHUBER, Jürgen: “Multi-column deep neural networks for image classification”. In: *2012*

- IEEE conference on computer vision and pattern recognition*. IEEE. 2012, pp. 3642–3649 (cit. on p. 1).
- [Con20] CONTRIBUTORS, MMsegmentation: MMsegmentation: Open-MMLab Semantic Segmentation Toolbox and Benchmark. <https://github.com/open-mmlab/msegmentation>. 2020 (cit. on p. 205).
- [Cor16] CORDTS, Marius; OMRAN, Mohamed; RAMOS, Sebastian; REHFELD, Timo; ENZWEILER, Markus; BENENSON, Rodrigo; FRANKE, Uwe; ROTH, Stefan and SCHIELE, Bernt: “The Cityscapes Dataset for Semantic Urban Scene Understanding”. In: *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016 (cit. on pp. 43, 65, 105, 139).
- [Cor19] CORDONNIER, Jean-Baptiste; LOUKAS, Andreas and JAGGI, Martin: “On the relationship between self-attention and convolutional layers”. In: *arXiv preprint arXiv:1911.03584* (2019) (cit. on p. 19).
- [Csi21] CSISZÁRIK, Adrián; KÓRÖSI-SZABÓ, Péter; MATSZANGOSZ, Ákos; PAPP, Gergely and VARGA, Dániel: “Similarity and Matching of Neural Network Representations”. In: *Advances in Neural Information Processing Systems*. Ed. by RANZATO, M.; BEYGEZIMER, A.; DAUPHIN, Y.; LIANG, P.S. and VAUGHAN, J. Wortman. Vol. 34. Curran Associates, Inc., 2021, pp. 5656–5668. URL: <https://proceedings.neurips.cc/paper/2021/file/2cb274e6ce940f47beb8011d8ecb1462-Paper.pdf> (cit. on pp. 8, 65–67, 70, 73, 75).
- [Cyb89] CYBENKO, George: “Approximation by superpositions of a sigmoidal function”. In: *Mathematics of control, signals and systems* 2.4 (1989), pp. 303–314 (cit. on p. 13).
- [Dai21] DAI, Zihang; LIU, Hanxiao; LE, Quoc V and TAN, Mingxing: “Coatnet: Marrying convolution and attention for all data sizes”. In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 3965–3977 (cit. on p. 19).

- [dAs21] D’ASCOLI, Stéphane; TOUVRON, Hugo; LEAVITT, Matthew L; MORCOS, Ari S; BIROLI, Giulio and SAGUN, Levent: “Convit: Improving vision transformers with soft convolutional inductive biases”. In: *International Conference on Machine Learning*. PMLR. 2021, pp. 2286–2296 (cit. on p. 19).
- [Dav22] DAVARI, MohammadReza; ASADI, Nader; MUDUR, Sudhir; ALJUNDI, Rahaf and BELILOVSKY, Eugene: “Probing Representation Forgetting in Supervised and Unsupervised Continual Learning”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2022, pp. 16712–16721 (cit. on pp. 42, 64, 65, 68, 72, 78, 94).
- [De 19] DE LANGE, Matthias; ALJUNDI, Rahaf; MASANA, Marc; PARISOT, Sarah; JIA, Xu; LEONARDIS, Ales; SLABAUGH, Gregory and TUYTELAARS, Tinne: “Continual learning: A comparative study on how to defy forgetting in classification tasks”. In: (2019), pp. 1–23. arXiv: 1909.08383. URL: <http://arxiv.org/abs/1909.08383> (cit. on pp. 24, 33–35, 43, 201, 203).
- [De 22] DE LANGE, Matthias; VEN, Gido van de and TUYTELAARS, Tinne: “Continual evaluation for lifelong learning: Identifying the stability gap”. In: *arXiv preprint arXiv:2205.13452* (2022) (cit. on pp. 42, 155).
- [Del21] DELANGE, Matthias; ALJUNDI, Rahaf; MASANA, Marc; PARISOT, Sarah; JIA, Xu; LEONARDIS, Ales; SLABAUGH, Greg and TUYTELAARS, Tinne: “A continual learning survey: Defying forgetting in classification tasks”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2021), pp. 1–1. DOI: 10.1109/TPAMI.2021.3057446 (cit. on p. 20).
- [Den09] DENG, Jia; DONG, Wei; SOCHER, Richard; LI, Li-Jia; LI, Kai and FEI-FEI, Li: “ImageNet: A large-scale hierarchical image database”. In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. 2009, pp. 248–255. DOI: 10.1109/CVPR.2009.5206848 (cit. on pp. 1, 43).

- [Dev18] DEVLIN, Jacob; CHANG, Ming-Wei; LEE, Kenton and TOUTANOVA, Kristina: “Bert: Pre-training of deep bidirectional transformers for language understanding”. In: *arXiv preprint arXiv:1810.04805* (2018) (cit. on p. 1).
- [Día18] DÍAZ-RODRÍGUEZ, Natalia; LOMONACO, Vincenzo; FILLIAT, David and MALTONI, Davide: “Don’t forget, there is more than forgetting: new metrics for Continual Learning”. In: *Nips* (2018). arXiv: 1810.13166. URL: <http://arxiv.org/abs/1810.13166> (cit. on p. 41).
- [Dos20] DOSOVITSKIY, Alexey; BEYER, Lucas; KOLESNIKOV, Alexander; WEISENBORN, Dirk; ZHAI, Xiaohua; UNTERTHINER, Thomas; DEGHANI, Mostafa; MINDERER, Matthias; HEIGOLD, Georg; GELLY, Sylvain et al.: “An image is worth 16x16 words: Transformers for image recognition at scale”. In: *arXiv preprint arXiv:2010.11929* (2020) (cit. on pp. 20, 135, 137, 140).
- [Dos21] DOSOVITSKIY, Alexey et al.: “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”. In: *ICLR* (2021) (cit. on pp. 18, 19, 77).
- [Dou20] DOUILLARD, Arthur; CORD, Matthieu; OLLION, Charles; ROBERT, Thomas and VALLE, Eduardo: “Podnet: Pooled outputs distillation for small-tasks incremental learning”. In: *Computer Vision—ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XX 16*. Springer. 2020, pp. 86–102 (cit. on p. 36).
- [Dou21a] DOUILLARD, Arthur; CHEN, Yifu; DAPOGNY, Arnaud and CORD, Matthieu: “PLOP: Learning without Forgetting for Continual Semantic Segmentation”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2021 (cit. on pp. 39, 87, 89).
- [Dou21b] DOUILLARD, Arthur; CHEN, Yifu; DAPOGNY, Arnaud and CORD, Matthieu: “Tackling Catastrophic Forgetting and Background Shift in Continual Semantic Segmentation”. In:

- CoRR abs/2106.15287 (2021). arXiv: 2106.15287. URL: <https://arxiv.org/abs/2106.15287> (cit. on pp. 40, 97).
- [Dou22] DOUILLARD, Arthur; RAMÉ, Alexandre; COUAIRO, Guillaume and CORD, Matthieu: “DyTox: Transformers for Continual Learning with DYnamic TOken eXpansion”. In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE. 2022, pp. 9275–9285 (cit. on p. 37).
- [Ebr20] EBRAHIMI, Sayna; MEIER, Franziska; CALANDRA, Roberto; DARRELL, Trevor and ROHRBACH, Marcus: “Adversarial continual learning”. In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16*. Springer. 2020, pp. 386–402 (cit. on p. 37).
- [Eve12] EVERINGHAM, M.; VAN GOOL, L.; WILLIAMS, C. K. I.; WINN, J. and ZISSERMAN, A.: The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. <http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html>. 2012 (cit. on pp. 86, 87).
- [Fer17] FERNANDO, Chrisantha; BANARSE, Dylan; BLUNDELL, Charles; ZWOLS, Yori; HA, David; RUSU, Andrei A.; PRITZEL, Alexander and WIERSTRA, Daan: “PathNet: Evolution Channels Gradient Descent in Super Neural Networks”. In: *CoRR abs/1701.08734* (2017). arXiv: 1701.08734. URL: <http://arxiv.org/abs/1701.08734> (cit. on p. 37).
- [Fra20] FRANKLE, Jonathan; DZIUGAITE, Gintare Karolina; ROY, Daniel M. and CARBIN, Michael: “Linear Mode Connectivity and the Lottery Ticket Hypothesis”. In: *Proceedings of the 37th International Conference on Machine Learning. ICML’20*. JMLR.org, 2020 (cit. on p. 42).
- [Fre94] FRENCH, Robert M.: “Catastrophic Forgetting in Connectionist Networks: Causes, Consequences and Solutions”. In: *Trends in Cognitive Sciences*. 1994, pp. 128–135 (cit. on p. 22).

- [Fuk80] FUKUSHIMA, Kunihiko: “Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position”. In: *Biological cybernetics* 36.4 (1980), pp. 193–202 (cit. on p. 14).
- [Gal21] GALLARDO, Jhair; HAYES, Tyler L. and KANAN, Christopher: “Self-Supervised Training Enhances Online Continual Learning”. In: *British Machine Vision Conference (BMVC)*. 2021 (cit. on pp. 88, 105, 106, 111, 129).
- [Gar22] GARG, Prachi; SALUJA, Rohit; BALASUBRAMANIAN, Vineeth N; ARORA, Chetan; SUBRAMANIAN, Anbumani and JAWAHAR, C.V.: “Multi-Domain Incremental Learning for Semantic Segmentation”. In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*. Jan. 2022, pp. 761–771 (cit. on p. 40).
- [Gei19] GEIRHOS, Robert; RUBISCH, Patricia; MICHAELIS, Claudio; BETHGE, Matthias; WICHMANN, Felix A and BRENDDEL, Wieland: “ImageNet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness.” In: *International Conference on Learning Representations*. 2019. URL: %5Curl%7Bhttps://openreview.net/forum?id=Bygh9j09KX%7D (cit. on p. 134).
- [Göt21] GÖTZ, Florian: The Data Deluge: What do we do with the data generated by AVs? | Polarion. en-US. Section: Learning Resources. Jan. 2021. URL: <https://blogs.sw.siemens.com/polarion/the-data-deluge-what-do-we-do-with-the-data-generated-by-avs/> (visited on 06/03/2023) (cit. on p. 6).
- [Has23] HASSANI, Ali; WALTON, Steven; LI, Jiachen; LI, Shen and SHI, Humphrey: “Neighborhood Attention Transformer”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2023, pp. 6185–6194 (cit. on pp. 137, 140, 205).

- [Hat20] HATAYA, Ryuichiro; ZDENEK, Jan; YOSHIKOE, Kazuki and NAKAYAMA, Hideki: “Faster AutoAugment: Learning Augmentation Strategies Using Backpropagation”. In: Glasgow, United Kingdom: Springer-Verlag, 2020, pp. 1–16. DOI: 10.1007/978-3-030-58595-2\_1. URL: %5Curl%7Bhttps://doi.org/10.1007/978-3-030-58595-2\_1%7D (cit. on p. 111).
- [Hay20] HAYES, Tyler L; KAFLE, Kushal; SHRESTHA, Robik; ACHARYA, Manoj and KANAN, Christopher: “REMIND Your Neural Network to Prevent Catastrophic Forgetting”. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2020 (cit. on p. 34).
- [He16] HE, Kaiming; ZHANG, Xiangyu; REN, Shaoqing and SUN, Jian: “Deep Residual Learning for Image Recognition”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2016 (cit. on pp. 14, 17, 137, 204).
- [Hin15] HINTON, Geoffrey; VINYALS, Oriol and DEAN, Jeffrey: “Distilling the Knowledge in a Neural Network”. In: *NIPS Deep Learning and Representation Learning Workshop*. 2015. URL: <http://arxiv.org/abs/1503.02531> (cit. on pp. 36, 38, 86).
- [Hor91] HORNIK, Kurt: “Approximation capabilities of multilayer feed-forward networks”. In: *Neural networks* 4.2 (1991), pp. 251–257 (cit. on p. 13).
- [Hou19] HOU, Saihui; PAN, Xinyu; LOY, Chen Change; WANG, Zilei and LIN, Dahua: “Learning a Unified Classifier Incrementally via Rebalancing”. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2019 (cit. on p. 36).
- [Hsu18] HSU, Yen-Chang; LIU, Yen-Cheng; RAMASAMY, Anita and KIRA, Zsolt: “Re-evaluating Continual Learning Scenarios: A Categorization and Case for Strong Baselines”. In: *NeurIPS Continual learning Workshop*. 2018. URL: <https://arxiv.org/abs/1810.12488> (cit. on pp. 21, 22, 35, 36, 43, 44, 52, 53, 55, 56, 85).

- [Iak19] IAKUBOVSKII, Pavel: Segmentation Models Pytorch. [https://github.com/qubvel/segmentation\\_models.pytorch](https://github.com/qubvel/segmentation_models.pytorch). 2019 (cit. on p. 204).
- [Iof15] IOFFE, Sergey and SZEGEDY, Christian: “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift”. In: *Proceedings of the 32nd International Conference on Machine Learning*. Ed. by BACH, Francis and BLEI, David. Vol. 37. Proceedings of Machine Learning Research. Lille, France: PMLR, 2015, pp. 448–456. URL: <https://proceedings.mlr.press/v37/ioffe15.html> (cit. on pp. 14, 16, 76).
- [Iof17] IOFFE, Sergey: “Batch renormalization: Towards reducing mini-batch dependence in batch-normalized models”. In: *Advances in neural information processing systems* 30 (2017) (cit. on p. 138).
- [Kal21] KALB, Tobias; ROSCHANI, Masoud; RUF, Miriam and BEYERER, Jürgen: “Continual Learning for Class- and Domain-Incremental Semantic Segmentation”. In: *2021 IEEE Intelligent Vehicles Symposium (IV)*. 2021, pp. 1345–1351. DOI: 10.1109/IV48863.2021.9575493 (cit. on pp. 7, 40, 44).
- [Kal22a] KALB, Tobias and BEYERER, Jürgen: “Causes of Catastrophic Forgetting in Class-Incremental Semantic Segmentation”. In: *Proceedings of the Asian Conference on Computer Vision (ACCV)*. Dec. 2022, pp. 56–73 (cit. on pp. 8, 63, 64, 66, 74, 78, 85).
- [Kal22b] KALB, Tobias; MAUTHE, Björn and BEYERER, Jürgen: “Improving Replay-Based Continual Semantic Segmentation with Smart Data Selection”. In: *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*. Macau, China: IEEE Press, 2022, pp. 1114–1121. DOI: 10.1109/ITSC55140.2022.9922284. URL: <https://doi.org/10.1109/ITSC55140.2022.9922284> (cit. on pp. 10, 44).
- [Kal23a] KALB, Tobias: “Measuring the Effects of Catastrophic Forgetting in Neural Networks”. In: *Proceedings of the 2022 Joint Workshop of Fraunhofer IOSB and Institute for Anthropomatics, Vision and Fusion Laboratory*. Joint Workshop of Fraunhofer IOSB and

- Institute for Anthropomatics, Vision and Fusion Laboratory. 2022 (Karlsruhe, Deutschland, July 31–Aug. 5, 2022). Vol. 62. *Karlsruher Schriften zur Anthropomatik / Lehrstuhl für Interaktive Echtzeitsysteme*, Karlsruhe Institut für Technologie ; Fraunhofer-Inst. für Optronik, Systemtechnik und Bildauswertung IOSB Karlsruhe. Karlsruhe Institut für Technologie (KIT), 2023, pp. 27–45 (cit. on pp. 8, 11, 63).
- [Kal23b] KALB, Tobias; AHUJA, Niket; ZHOU, Jingxing and BEYERER, Jürgen: “Effects of Architectures on Continual Semantic Segmentation”. In: *2023 IEEE Intelligent Vehicles Symposium (IV)*. 2023, pp. 1–8. DOI: 10.1109/IV55152.2023.10186597 (cit. on p. 9).
- [Kal23c] KALB, Tobias and BEYERER, Jürgen: “Principles of Forgetting in Domain-Incremental Semantic Segmentation in Adverse Weather Conditions”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2023, pp. 19508–19518 (cit. on pp. 9, 85).
- [Kar20] KARPATY, Andrej: Andrej Karpathy - AI for Full-Self Driving at Tesla. 2020. URL: <https://www.youtube.com/watch?v=hx7BXih7zx8%5C&t=886s> (visited on 05/29/2023) (cit. on p. 6).
- [Kim20] KIM, Joonhyuk; YOO, Sahng-Min; PARK, Gyeong-Moon and KIM, Jong-Hwan: Continual Unsupervised Domain Adaptation for Semantic Segmentation. 2020. DOI: 10.48550/ARXIV.2010.09236. URL: <https://arxiv.org/abs/2010.09236> (cit. on p. 41).
- [Kir15] KIRKPATRICK, James; PASCANU, Razvan; RABINOWITZ, Neil; VENESS, Joel and DESJARDINS, Guillaume: “Overcoming catastrophic forgetting in neural networks”. In: *Proceedings of the National Academy of Sciences* 114.13 (Mar. 2015), pp. 3521–3526. DOI: 10.1073/pnas.1611835114. arXiv: arXiv:1612.00796v2 [cs.LG] (cit. on pp. 35, 36, 48, 65, 80, 88, 89, 92, 96, 99, 103, 199).
- [Kir23] KIRILLOV, Alexander et al.: Segment Anything. 2023. arXiv: 2304.02643 [cs.CV] (cit. on p. 6).

- [Kle23] KLEINMAN, Michael; ACHILLE, Alessandro and SOATTO, Stefano: “Critical Learning Periods for Multisensory Integration in Deep Networks”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 24296–24305 (cit. on p. 129).
- [Kli20] KLINGNER, Marvin; BÄR, Andreas; DONN, Philipp and FINGSCHEIDT, Tim: “Class-incremental learning for semantic segmentation re-using neither old data nor old labels”. In: *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*. IEEE. 2020, pp. 1–8 (cit. on pp. 39, 46–48, 50, 88).
- [Kor19] KORNBLITH, Simon; NOROUZI, Mohammad; LEE, Honglak and HINTON, Geoffrey: “Similarity of Neural Network Representations Revisited”. In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by CHAUDHURI, Kamalika and SALAKHUTDINOV, Ruslan. Vol. 97. Proceedings of Machine Learning Research. PMLR, 2019, pp. 3519–3529. URL: <https://proceedings.mlr.press/v97/kornblith19a.html> (cit. on pp. 64, 66, 72).
- [Kor21] KORNBLITH, Simon; CHEN, Ting; LEE, Honglak and NOROUZI, Mohammad: “Why do better loss functions lead to less transferable features?” In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 28648–28662 (cit. on p. 146).
- [Kri09] KRIZHEVSKY, Alex; HINTON, Geoffrey et al.: “Learning multiple layers of features from tiny images”. In: (2009) (cit. on p. 43).
- [LeC89] LECUN, Yann; BOSER, Bernhard; DENKER, John S; HENDERSON, Donnie; HOWARD, Richard E; HUBBARD, Wayne and JACKEL, Lawrence D: “Backpropagation applied to handwritten zip code recognition”. In: *Neural computation* 1.4 (1989), pp. 541–551 (cit. on pp. 1, 14).
- [Len15] LENC, Karel and VEDALDI, Andrea: “Understanding Image Representations by Measuring Their Equivariance and Equivalence”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2015 (cit. on p. 67).

- [Les20] LESORT, Timothée; STOIAN, Andrei and FILLIAT, David: Regularization Shortcomings for Continual Learning. 2020. arXiv: 1912.03049 [cs.LG] (cit. on pp. 36, 94).
- [Les21] LESORT, Timothée; GEORGE, Thomas and RISH, Irina: Continual Learning in Deep Networks: an Analysis of the Last Layer. 2021. DOI: 10.48550/ARXIV.2106.01834. URL: <https://arxiv.org/abs/2106.01834> (cit. on p. 98).
- [Les22] LESORT, Timothée: “Continual Feature Selection: Spurious Features in Continual Learning”. In: *arXiv preprint arXiv:2203.01012* (2022) (cit. on p. 23).
- [Li18] LI, Zhizhong and HOIEM, Derek: “Learning without Forgetting”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40.12 (2018), pp. 2935–2947. DOI: 10.1109/TPAMI.2017.2773081. arXiv: 1606.09282 (cit. on pp. 36, 38, 48, 65, 86, 88, 89, 92, 96, 99, 103, 199).
- [Li21] LI, Zhuoyun; ZHONG, Changhong; LIU, Sijia; WANG, Ruixuan and ZHENG, Wei-Shi: “Preserving earlier knowledge in continual learning with the help of all previous feature extractors”. In: *arXiv preprint arXiv:2104.13614* (2021) (cit. on p. 37).
- [Li22a] LI, Aodi; ZHUANG, Liansheng; FAN, Shuo and WANG, Shafei: “Learning Common and Specific Visual Prompts for Domain Generalization”. In: *Proceedings of the Asian Conference on Computer Vision (ACCV)*. Dec. 2022, pp. 4260–4275 (cit. on p. 134).
- [Li22b] LI, Duo; CAO, Guimei; XU, Yunlu; CHENG, Zhazhan and NIU, Yi: Technical Report for ICCV 2021 Challenge SSLAD-Track3B: Transformers Are Better Continual Learners. 2022. DOI: 10.48550/ARXIV.2201.04924. URL: <https://arxiv.org/abs/2201.04924> (cit. on pp. 135, 137).
- [Liu21] LIU, Ze; LIN, Yutong; CAO, Yue; HU, Han; WEI, Yixuan; ZHANG, Zheng; LIN, Stephen and GUO, Baining: “Swin Transformer: Hierarchical Vision Transformer Using Shifted Windows”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2021, pp. 10012–10022 (cit. on p. 137).

- [Liu22] LIU, Zhuang; MAO, Hanzi; WU, Chao-Yuan; FEICHTENHOFER, Christoph; DARRELL, Trevor and XIE, Saining: “A ConvNet for the 2020s”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2022)* (cit. on pp. 137, 148).
- [Lom20] LOMONACO, Vincenzo; MALTONI, Davide and PELLEGRINI, Lorenzo: “Rehearsal-Free Continual Learning Over Small Non-I.I.D. Batches”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops. June 2020* (cit. on pp. 76, 116).
- [Lon15] LONG, Jonathan; SHELHAMER, Evan and DARRELL, Trevor: “Fully Convolutional Networks for Semantic Segmentation”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2015 (cit. on pp. 1, 27, 28).
- [Lop17] LOPEZ-PAZ, David and RANZATO, Marc’Aurelio: Gradient Episodic Memory for Continual Learning. June 26, 2017. URL: <http://arxiv.org/pdf/1706.08840v5> (cit. on p. 34).
- [Los19] LOSHCILOV, Ilya and HUTTER, Frank: “Decoupled Weight Decay Regularization”. In: *International Conference on Learning Representations*. 2019. URL: <https://openreview.net/forum?id=Bkg6RiCqY7%7D> (cit. on p. 139).
- [Lu17] LU, Zhou; PU, Hongming; WANG, Feicheng; HU, Zhiqiang and WANG, Liwei: “The Expressive Power of Neural Networks: A View from the Width”. In: *Advances in Neural Information Processing Systems*. Ed. by GUYON, I.; LUXBURG, U. Von; BENGIO, S.; WALLACH, H.; FERGUS, R.; VISHWANATHAN, S. and GARNETT, R. Vol. 30. Curran Associates, Inc., 2017. URL: [https://proceedings.neurips.cc/paper\\_files/paper/2017/file/32cbf687880eb1674a07bf17761dd3a-Paper.pdf](https://proceedings.neurips.cc/paper_files/paper/2017/file/32cbf687880eb1674a07bf17761dd3a-Paper.pdf) (cit. on p. 13).

- [Mal18] MALLYA, Arun and LAZEBNIK, Svetlana: “Packnet: Adding multiple tasks to a single network by iterative pruning”. In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2018, pp. 7765–7773 (cit. on p. 37).
- [Mar21] MARACANI, Andrea; MICHELI, Umberto; TOLDO, Marco and ZANUTTIGH, Pietro: “RECALL: Replay-Based Continual Learning in Semantic Segmentation”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2021, pp. 7026–7035 (cit. on pp. 40, 89).
- [Mar22] MARSDEN, Robert A.; WIEWEL, Felix; DÖBLER, Mario; YANG, Yang and YANG, Bin: Continual Unsupervised Domain Adaptation for Semantic Segmentation using a Class-Specific Transfer. 2022. DOI: 10.48550/ARXIV.2208.06507. URL: <https://arxiv.org/abs/2208.06507> (cit. on pp. 40, 41).
- [Mas20] MASANA, Marc; LIU, Xialei; TWARDOWSKI, Bartłomiej; MENTA, Mikel; BAGDANOV, Andrew D and WEIJER, Joost van de: “Class-incremental learning: survey and performance evaluation”. In: *arXiv preprint arXiv:2010.15277* (2020) (cit. on pp. 23, 36, 43, 44, 85, 86, 97).
- [McC89] McCLOSKEY, Michael and COHEN, Neal J.: “Catastrophic Interference in Connectionist Networks: The Sequential Learning Problem”. English (US). In: *Psychology of Learning and Motivation - Advances in Research and Theory* 24.C (Jan. 1, 1989), pp. 109–165. DOI: 10.1016/S0079-7421(08)60536-8 (cit. on pp. 2, 22).
- [McI18] McINNES, Leland; HEALY, John and MELVILLE, James: UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. Feb. 9, 2018. URL: <http://arxiv.org/pdf/1802.03426v3> (cit. on p. 57).
- [Meh21] MEHTA, Sanket Vaibhav; PATIL, Darshan; CHANDAR, Sarath and STRUBELL, Emma: An Empirical Investigation of the Role of Pre-training in Lifelong Learning. 2021. DOI: 10.48550/ARXIV.2112.

09153. URL: <https://arxiv.org/abs/2112.09153> (cit. on pp. 88, 105, 106, 111).
- [Mic19] MICHIELI, Umberto and ZANUTTIGH, Pietro: “Incremental learning techniques for semantic segmentation”. In: *Proceedings - 2019 International Conference on Computer Vision Workshop, ICCVW 2019* (July 2019), pp. 3205–3212. doi: 10.1109/ICCVW.2019.00400. arXiv: 1907.13372. URL: <http://arxiv.org/abs/1907.13372> (cit. on pp. 39, 47, 65, 87).
- [Mic21] MICHIELI, Umberto and ZANUTTIGH, Pietro: “Continual Semantic Segmentation via Repulsion-Attraction of Sparse and Disentangled Latent Representations”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2021, pp. 1114–1124 (cit. on pp. 39, 87, 89).
- [Mir20] MIRZADEH, Seyed Iman; FARAJTABAR, Mehrdad; PASCANU, Razvan and GHASEMZADEH, Hassan: “Understanding the Role of Training Regimes in Continual Learning”. In: *Advances in Neural Information Processing Systems*. Ed. by LAROCHELLE, H.; RANZATO, M.; HADSELL, R.; BALCAN, M. F. and LIN, H. Vol. 33. Curran Associates, Inc., 2020, pp. 7308–7320. URL: <https://proceedings.neurips.cc/paper/2020/file/518a38cc9a0173d0b2dc088166981cf8-Paper.pdf> (cit. on p. 104).
- [Mir21] MIRZADEH, Seyed Iman; FARAJTABAR, Mehrdad; GORUR, Dilan; PASCANU, Razvan and GHASEMZADEH, Hassan: “Linear Mode Connectivity in Multitask and Continual Learning”. In: *International Conference on Learning Representations*. 2021. URL: [https://openreview.net/forum?id=Fmg\\_fQYUejf](https://openreview.net/forum?id=Fmg_fQYUejf) (cit. on p. 42).
- [Mir22a] MIRZA, M Jehanzeb; MASANA, Marc; POSSEGGGER, Horst and BISCHOF, Horst: “An Efficient Domain-Incremental Learning Approach to Drive in All Weather Conditions”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 3001–3011 (cit. on p. 40).

- [Mir22b] MIRZADEH, Seyed Iman; CHAUDHRY, Arslan; YIN, Dong; NGUYEN, Timothy; PASCANU, Razvan; GORUR, Dilan and FARAJTABAR, Mehrdad: Architecture Matters in Continual Learning. 2022. DOI: 10 . 48550 / ARXIV . 2202 . 00275. URL: <https://arxiv.org/abs/2202.00275> (cit. on pp. 30, 106, 125, 134).
- [Mun21] MUNDT, Martin; PLIUSHCH, Iuliia and RAMESH, Visvanathan: “Neural Architecture Search of Deep Priors: Towards Continual Learning Without Catastrophic Interference”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. June 2021, pp. 3523–3532 (cit. on p. 134).
- [Mur20] MURATA, Kengo; TOYOTA, Tetsuya and OHARA, Kouzou: “What is happening inside a continual learning model? - A representation-based evaluation of representational forgetting - A R”. In: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*. Vol. 2020-June. 2020, pp. 952–956. DOI: 10 . 1109 / CVPRW50498 . 2020 . 00125 (cit. on pp. 41, 65, 75, 77).
- [Nas21] NASEER, Muzammal; RANASINGHE, Kanchana; KHAN, Salman; HAYAT, Munawar; KHAN, Fahad and YANG, Ming-Hsuan: “Intriguing Properties of Vision Transformers”. In: *Advances in Neural Information Processing Systems*. 2021 (cit. on p. 134).
- [Ney20] NEYSHABUR, Behnam; SEDGHI, Hanie and ZHANG, Chiyuan: “What is being transferred in transfer learning?” In: *Advances in Neural Information Processing Systems*. Ed. by LAROCHELLE, H.; RANZATO, M.; HADSELL, R.; BALKAN, M.F. and LIN, H. Vol. 33. Curran Associates, Inc., 2020, pp. 512–523. URL: <https://proceedings.neurips.cc/paper/2020/file/0607f4c705595b911a4f3e7a127b44e0-Paper.pdf> (cit. on pp. 64, 79, 94, 104, 111).
- [Ngu19] NGUYEN, Cuong V.; ACHILLE, Alessandro; LAM, Michael; HASSNER, Tal; MAHADEVAN, Vijay and SOATTO, Stefano: “Toward Understanding Catastrophic Forgetting in Continual Learning”. In:

- (Aug. 2019). DOI: 10.48550/ARXIV.1908.01091. arXiv: 1908.01091 [cs.LG]. URL: <http://arxiv.org/abs/1908.01091> (cit. on pp. 42, 127).
- [Ola17] OLAH, Chris; MORDVINTSEV, Alexander and SCHUBERT, Ludwig: “Feature Visualization”. In: *Distill* (2017). <https://distill.pub/2017/feature-visualization>. DOI: 10.23915/distill.00007 (cit. on pp. 18, 122).
- [Pau22] PAUL, Sayak and CHEN, Pin-Yu: “Vision Transformers are Robust Learners”. In: *Proceedings of the AAAI Conference on Artificial Intelligence* (2022) (cit. on p. 134).
- [Pha22] PHAM, Quang; LIU, Chenghao and STEVEN, HOI: “Continual Normalization: Rethinking Batch Normalization for Online Continual Learning”. In: *International Conference on Learning Representations*. 2022 (cit. on pp. 17, 118, 138).
- [Rad21] RADFORD, Alec; KIM, Jong Wook; HALLACY, Chris; RAMESH, Aditya; GOH, Gabriel; AGARWAL, Sandhini; SASTRY, Girish; ASKELL, Amanda; MISHKIN, Pamela; CLARK, Jack et al.: “Learning transferable visual models from natural language supervision”. In: *International conference on machine learning*. PMLR. 2021, pp. 8748–8763 (cit. on p. 155).
- [Rag21] RAGHU, Maithra; UNTERTHINER, Thomas; KORNBLITH, Simon; ZHANG, Chiyuan and DOSOVITSKIY, Alexey: “Do vision transformers see like convolutional neural networks?” In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 12116–12128 (cit. on pp. 19, 134, 140).
- [Ram21] RAMASESH, Vinay Venkatesh; DYER, Ethan and RAGHU, Maithra: “Anatomy of Catastrophic Forgetting: Hidden Representations and Task Semantics”. In: *International Conference on Learning Representations*. 2021. URL: <https://openreview.net/forum?id=LhY8QdUGSuw> (cit. on pp. 42, 64, 68, 72).
- [Reb17] REBUFFI, Sylvestre-Alvise; KOLESNIKOV, Alexander; SPERL, Georg and LAMPERT, Christoph H.: “iCaRL: Incremental Classifier and Representation Learning”. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2017, pp. 5533–5542. DOI: 10.1109/CVPR.2017.587 (cit. on pp. 34, 36).

- [Rom18] ROMERA, Eduardo; ÁLVAREZ, José M.; BERGASA, Luis M. and ARROYO, Roberto: “ERFNet: Efficient Residual Factorized ConvNet for Real-Time Semantic Segmentation”. In: *IEEE Transactions on Intelligent Transportation Systems* 19.1 (2018), pp. 263–272. DOI: 10.1109/TITS.2017.2750080 (cit. on pp. 27, 48, 50, 65, 88, 124–126, 204).
- [Rom22] ROMBACH, Robin; BLATTMANN, Andreas; LORENZ, Dominik; ESSER, Patrick and OMMER, Björn: “High-resolution image synthesis with latent diffusion models”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2022*, pp. 10684–10695 (cit. on p. 155).
- [Ron15] RONNEBERGER, Olaf; FISCHER, Philipp and BROX, Thomas: U-Net: Convolutional Networks for Biomedical Image Segmentation. 2015. DOI: 10.48550/ARXIV.1505.04597. URL: <https://arxiv.org/abs/1505.04597> (cit. on pp. 27, 88).
- [Rum86] RUMELHART, David E; HINTON, Geoffrey E and WILLIAMS, Ronald J: “Learning representations by back-propagating errors”. In: *nature* 323.6088 (1986), pp. 533–536 (cit. on p. 12).
- [Sak21] SAKARIDIS, Christos; DAI, Dengxin and VAN GOOL, Luc: “ACDC: The Adverse Conditions Dataset with Correspondences for Semantic Driving Scene Understanding”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2021 (cit. on pp. 65, 105, 107, 139).
- [Sal16] SALIMANS, Tim and KINGMA, Durk P: “Weight Normalization: A Simple Reparameterization to Accelerate Training of Deep Neural Networks”. In: *Advances in Neural Information Processing Systems*. Ed. by LEE, D.; SUGIYAMA, M.; LUXBURG, U.; GUYON, I. and GARNETT, R. Vol. 29. Curran Associates, Inc., 2016. URL: <https://proceedings.neurips.cc/paper/2016/file/ed265bc903a5a097f61d3ec064d96d2e-Paper.pdf> (cit. on p. 98).

- [Sch23a] SCHWONBERG, Manuel; EL BOUZATI, Fadoua; SCHMIDT, Nico M and GOTTSCHALK, Hanno: “Augmentation-based Domain Generalization for Semantic Segmentation”. In: *2023 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2023, pp. 1–8 (cit. on p. 105).
- [Sch23b] SCHWONBERG, Manuel; NIEMEIJER, Joshua; TERMÖHLEN, Jan-Aike; SCHÄFER, Jörg P; SCHMIDT, Nico M; GOTTSCHALK, Hanno and FINGSCHIEDT, Tim: “Survey on Unsupervised Domain Adaptation for Semantic Segmentation for Visual Perception in Automated Driving”. In: *IEEE Access* (2023) (cit. on p. 41).
- [Shi17] SHIN, Hanul; LEE, Jung Kwon; KIM, Jaehong and KIM, Jiwon: “Continual Learning with Deep Generative Replay”. In: *Advances in Neural Information Processing Systems*. Ed. by GUYON, I.; LUXBURG, U. V.; BENGIO, S.; WALLACH, H.; FERGUS, R.; VISHWANATHAN, S. and GARNETT, R. Vol. 30. Curran Associates, Inc., 2017. URL: <https://proceedings.neurips.cc/paper/2017/file/0efbe98067c6c73dba1250d2beaa81f9-Paper.pdf> (cit. on p. 34).
- [Smi23] SMITH, James Seale; KARLINSKY, Leonid; GUTTA, Vyshnavi; CASCANTE-BONILLA, Paola; KIM, Donghyun; ARBELLE, Assaf; PANDA, Rameswar; FERIS, Rogerio and KIRA, Zsolt: “CODA-Prompt: COntinual Decomposed Attention-Based Prompting for Rehearsal-Free Continual Learning”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2023, pp. 11909–11919 (cit. on p. 156).
- [Tas19] TASAR, Onur; TARABALKA, Yuliya and ALLIEZ, Pierre: “Incremental Learning for Semantic Segmentation of Large-Scale Remote Sensing Data”. In: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 12.9 (2019), pp. 3524–3537. DOI: 10.1109/JSTARS.2019.2925416. arXiv: 1810.12448 (cit. on pp. 38, 48, 49, 65).
- [Ter21] TERMÖHLEN, Jan-Aike; KLINGNER, Marvin; BRETTIN, Leon J.; SCHMIDT, Nico M. and FINGSCHIEDT, Tim: “Continual Unsupervised Domain Adaptation for Semantic Segmentation by

- Online Frequency Domain Style Transfer”. In: *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*. 2021, pp. 2881–2888. DOI: 10.1109/ITSC48978.2021.9564566 (cit. on p. 41).
- [Tou21] TOUVRON, Hugo; CORD, Matthieu; DOUZE, Matthijs; MASSA, Francisco; SABLAYROLLES, Alexandre and JÉGOU, Hervé: “Training data-efficient image transformers & distillation through attention”. In: *International conference on machine learning*. PMLR. 2021, pp. 10347–10357 (cit. on p. 27).
- [Uhl22] UHLEMAYER, Svenja; ROTTMANN, Matthias and GOTTSCHALK, Hanno: “Towards unsupervised open world semantic segmentation”. In: *Uncertainty in Artificial Intelligence*. PMLR. 2022, pp. 1981–1991 (cit. on p. 40).
- [Uly16] ULYANOV, Dmitry; VEDALDI, Andrea and LEMPITSKY, Victor: “Instance normalization: The missing ingredient for fast stylization”. In: *arXiv preprint arXiv:1607.08022* (2016) (cit. on p. 138).
- [Van18] VAN DE VEN, Gido M and TOLIAS, Andreas S: “Generative replay with feedback connections as a general strategy for continual learning”. In: *arXiv preprint arXiv:1809.10635* (2018) (cit. on pp. 21, 36).
- [Vas17] VASWANI, Ashish; SHAZEER, Noam; PARMAR, Niki; USZKOREIT, Jakob; JONES, Llion; GOMEZ, Aidan N; KAISER, Łukasz and POLOSUKHIN, Illia: “Attention is All you Need”. In: *Advances in Neural Information Processing Systems*. Ed. by GUYON, I.; LUXBURG, U. Von; BENGIO, S.; WALLACH, H.; FERGUS, R.; VISHWANATHAN, S. and GARNETT, R. Vol. 30. Curran Associates, Inc., 2017. URL: %5Curl%7Bhttps://proceedings.neurips.cc/paper\_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf%7D (cit. on p. 18).
- [Ven20] VEN, Gido M. van de; SIEGELMANN, Hava T. and TOLIAS, Andreas S.: “Brain-inspired replay for continual learning with artificial neural networks”. In: *Nature Communications* 11.1 (2020). DOI: 10.1038/s41467-020-17866-2 (cit. on pp. 35, 56).

- [Ven22] VEN, Gido M van de; TUYTELAARS, Tinne and TOLIAS, Andreas S: “Three types of incremental learning”. In: *Nature Machine Intelligence* (2022), pp. 1–13 (cit. on p. 21).
- [Ver21] VERWIMP, Eli; DE LANGE, Matthias and TUYTELAARS, Tinne: “Rehearsal Revealed: The Limits and Merits of Revisiting Samples in Continual Learning”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2021, pp. 9385–9394 (cit. on pp. 35, 54).
- [Vij22] VIJAYENTHIRAN, Viknesh: Cruise opens up driverless taxi service to public in San Francisco. Feb. 2022. URL: %5Curl%7Bhttps://www.motorauthority.com/news/1132494\_cruise-opens-up-driverless-taxi-service-to-public-in-san-francisco%7D (visited on 06/03/2023) (cit. on p. 2).
- [Wan19] WANG, Jingdong et al.: “Deep High-Resolution Representation Learning for Visual Recognition”. In: *TPAMI* (2019) (cit. on pp. 88, 125, 127).
- [Wan20] WANG, Haohan; WU, Xindi; HUANG, Zeyi and XING, Eric P.: “High-Frequency Component Helps Explain the Generalization of Convolutional Neural Networks”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2020 (cit. on p. 109).
- [Wan22a] WANG, Zeyu; BAI, Yutong; ZHOU, Yuyin and XIE, Cihang: “Can CNNs Be More Robust Than Transformers?” In: *arXiv preprint arXiv:2206.03452* (2022) (cit. on pp. 134, 148, 149).
- [Wan22b] WANG, Zifeng; ZHANG, Zizhao; EBRAHIMI, Sayna; SUN, Ruoxi; ZHANG, Han; LEE, Chen-Yu; REN, Xiaoqi; SU, Guolong; PEROT, Vincent; DY, Jennifer et al.: “Dualprompt: Complementary prompting for rehearsal-free continual learning”. In: *European Conference on Computer Vision*. Springer. 2022, pp. 631–648 (cit. on p. 156).

- [Wan22c] WANG, Zifeng; ZHANG, Zizhao; LEE, Chen-Yu; ZHANG, Han; SUN, Ruoxi; REN, Xiaoqi; SU, Guolong; PEROT, Vincent; DY, Jennifer and PFISTER, Tomas: “Learning to prompt for continual learning”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 139–149 (cit. on p. 156).
- [Wel09] WELLING, Max: “Herding dynamical weights to learn”. In: *Proceedings of the 26th Annual International Conference on Machine Learning - ICML '09*. Ed. by DANYLUK, Andrea; BOTTOU, Léon and LITTMAN, Michael. New York, New York, USA: ACM Press, 2009, pp. 1–8. DOI: 10.1145/1553374.1553517 (cit. on p. 34).
- [Wol20] WOLF, Thomas et al.: “Transformers: State-of-the-Art Natural Language Processing”. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Online: Association for Computational Linguistics, Oct. 2020, pp. 38–45. URL: %5Curl%7Bhttps://www.aclweb.org/anthology/2020.emnlp-demos.6%7D (cit. on p. 204).
- [Wor20] WORTSMAN, Mitchell; RAMANUJAN, Vivek; LIU, Rosanne; KEMBHAVI, Aniruddha; RASTEGARI, Mohammad; YOSINSKI, Jason and FARHADI, Ali: “Supermasks in superposition”. In: *Advances in Neural Information Processing Systems 33* (2020), pp. 15173–15184 (cit. on p. 38).
- [Wu18a] WU, Chenshe; HERRANZ, Luis; LIU, Xialei; WANG, Yaxing; WEIJER, Joost van de and RADUCANU, Bogdan: “Memory Replay GANs: learning to generate images from new categories without forgetting”. In: *Conference on Neural Information Processing Systems (NIPS)*. 2018 (cit. on p. 34).
- [Wu18b] WU, Yuxin and HE, Kaiming: “Group normalization”. In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 3–19 (cit. on pp. 17, 138).
- [Wu19a] WU, Yue; CHEN, Yinpeng; WANG, Lijuan; YE, Yuancheng; LIU, Zicheng; GUO, Yandong and FU, Yun: Large Scale Incremental

- Learning. May 30, 2019. URL: <http://arxiv.org/pdf/1905.13260v1> (cit. on pp. 34, 36).
- [Wu19b] WU, Zuxuan; WANG, Xin; GONZALEZ, Joseph E.; GOLDSTEIN, Tom and DAVIS, Larry S.: “ACE: Adapting to Changing Environments for Semantic Segmentation”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2019 (cit. on p. 41).
- [Xia18] XIAO, Tete; LIU, Yingcheng; ZHOU, Bolei; JIANG, Yuning and SUN, Jian: “Unified Perceptual Parsing for Scene Understanding”. In: *European Conference on Computer Vision*. Springer. 2018 (cit. on p. 137).
- [Xie21] XIE, Enze; WANG, Wenhai; YU, Zhiding; ANANDKUMAR, Anima; ALVAREZ, Jose M and LUO, Ping: “SegFormer: Simple and efficient design for semantic segmentation with transformers”. In: *Advances in Neural Information Processing Systems 34 (2021)*, pp. 12077–12090 (cit. on pp. 27, 123, 124, 134, 137, 204).
- [Xu18] XU, Ju and ZHU, Zhanxing: “Reinforced Continual Learning”. In: *Advances in Neural Information Processing Systems*. Vol. 31. 2018. URL: <https://proceedings.neurips.cc/paper/2018/file/cee631121c2ec9232f3a2f028ad5c89b-Paper.pdf> (cit. on p. 134).
- [Yan21] YAN, Shipeng; XIE, Jiangwei and HE, Xuming: “Der: Dynamically expandable representation for class incremental learning”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 3014–3023 (cit. on p. 37).
- [Yu20] YU, Fisher; CHEN, Haofeng; WANG, Xin; XIAN, Wenqi; CHEN, Yingying; LIU, Fangchen; MADHAVAN, Vashisht and DARRELL, Trevor: “BDD100K: A Diverse Driving Dataset for Heterogeneous Multitask Learning”. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2020 (cit. on p. 44).
- [Yu21] YU, Changqian; GAO, Changxin; WANG, Jingbo; YU, Gang; SHEN, Chunhua and SANG, Nong: “Bisenet v2: Bilateral network with guided aggregation for real-time semantic segmentation”. In:

- International Journal of Computer Vision* 129 (2021), pp. 3051–3068 (cit. on pp. 88, 125, 126).
- [Yu22] YU, Lu; LIU, Xialei and WEIJER, Joost van de: “Self-Training for Class-Incremental Semantic Segmentation”. In: *IEEE Transactions on Neural Networks and Learning Systems* (2022), pp. 1–12. DOI: 10.1109/TNNLS.2022.3155746 (cit. on p. 40).
- [Zbo21] ZBONTAR, Jure; JING, Li; MISRA, Ishan; LECUN, Yann and DENY, Stéphane: “Barlow twins: Self-supervised learning via redundancy reduction”. In: *International Conference on Machine Learning*. PMLR. 2021, pp. 12310–12320 (cit. on pp. 111, 204).
- [Zei14] ZEILER, Matthew D and FERGUS, Rob: “Visualizing and understanding convolutional networks”. In: *Computer Vision—ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6–12, 2014, Proceedings, Part I 13*. Springer. 2014, pp. 818–833 (cit. on p. 17).
- [Zen17] ZENKE, Friedemann; POOLE, Ben and GANGULI, Surya: “Continual Learning Through Synaptic Intelligence”. In: *Proceedings of the 34th International Conference on Machine Learning*. Ed. by PRECUP, Doina and TEH, Yee Whye. Vol. 70. Proceedings of Machine Learning Research. International Convention Centre, Sydney, Australia: PMLR, 2017, pp. 3987–3995. URL: <http://proceedings.mlr.press/v70/zenke17a.html> (cit. on pp. 35, 36).
- [Zha17] ZHAO, Hengshuang et al.: “Pyramid Scene Parsing Network”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. July 2017 (cit. on p. 27).
- [Zha19a] ZHAI, Mengyao; CHEN, Lei; TUNG, Frederick; HE, Jiawei; NAWHAL, Megha and MORI, Greg: “Lifelong GAN: Continual Learning for Conditional Image Generation”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2019 (cit. on p. 34).
- [Zha19b] ZHANG, Chiyuan; BENGIO, Samy and SINGER, Yoram: “Are All Layers Created Equal? 2019”. DOI: 10.48550/ARXIV.1902.01996. URL: <https://arxiv.org/abs/1902.01996> (cit. on p. 117).

- [Zha22] ZHANG, Chongzhi; ZHANG, Mingyuan; ZHANG, Shanghang; JIN, Daisheng; ZHOU, Qiang; CAI, Zhongang; ZHAO, Haiyu; LIU, Xi-anlong and LIU, Ziwei: “Delving Deep Into the Generalization of Vision Transformers Under Distribution Shifts”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2022, pp. 7277–7286 (cit. on p. 134).
- [Zho19] ZHOU, Peng; MAI, Long; ZHANG, Jianming; XU, Ning; WU, Zuxuan and DAVIS, Larry S: “M2kd: Multi-model and multi-level knowledge distillation for incremental learning”. In: *arXiv preprint arXiv:1904.01769* (2019) (cit. on p. 36).

## Publications

- [1] KALB, Tobias; ROSCHANI, Masoud; RUF, Miriam and BEYERER, Jürgen: “Continual Learning for Class- and Domain-Incremental Semantic Segmentation”. In: *2021 IEEE Intelligent Vehicles Symposium (IV)*. 2021, pp. 1345–1351. DOI: 10.1109/IV48863.2021.9575493.
- [2] KALB, Tobias and BEYERER, Jürgen: “Causes of Catastrophic Forgetting in Class-Incremental Semantic Segmentation”. In: *Proceedings of the Asian Conference on Computer Vision (ACCV)*. Dec. 2022, pp. 56–73.
- [3] KALB, Tobias; MAUTHE, Björn and BEYERER, Jürgen: “Improving Replay-Based Continual Semantic Segmentation with Smart Data Selection”. In: *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*. Macau, China: IEEE Press, 2022, pp. 1114–1121. DOI: 10.1109/ITSC55140.2022.9922284. URL: <https://doi.org/10.1109/ITSC55140.2022.9922284>.
- [4] KALB, Tobias: “Measuring the Effects of Catastrophic Forgetting in Neural Networks”. In: *Proceedings of the 2022 Joint Workshop of Fraunhofer IOSB and Institute for Anthropomatics, Vision and Fusion Laboratory*. Joint Workshop of Fraunhofer IOSB and Institute for Anthropomatics, Vision and Fusion Laboratory. 2022 (Karlsruhe, Deutschland, July 31–Aug. 5, 2022). Vol. 62. *Karlsruher Schriften zur Anthropomatik / Lehrstuhl für Interaktive Echtzeitsysteme*, Karlsruher Institut für Technologie ; Fraunhofer-Inst. für Optronik, Systemtechnik und Bildauswertung IOSB Karlsruhe. Karlsruher Institut für Technologie (KIT), 2023, pp. 27–45.
- [5] KALB, Tobias; AHUJA, Niket; ZHOU, Jingxing and BEYERER, Jürgen: “Effects of Architectures on Continual Semantic Segmentation”. In:

*2023 IEEE Intelligent Vehicles Symposium (IV)*. 2023, pp. 1–8. DOI: 10.1109/IV55152.2023.10186597.

- [6] KALB, Tobias and BEYERER, Jürgen: “Principles of Forgetting in Domain-Incremental Semantic Segmentation in Adverse Weather Conditions”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2023, pp. 19508–19518.

# List of Figures

1.1	Illustration of a typical sensor suite for an exemplary automated vehicle. . . . .	3
2.1	Artificial Neural Network with two hidden layers. . . . .	12
2.2	Illustration of convolution and a sequential max pooling operation. . . . .	15
2.3	Illustration of different normalization methods. . . . .	17
2.4	Feature Visualization from different layers of a CNN trained on ImageNet . . . . .	18
2.5	Overview of the Vision Transformer architecture . . . . .	20
2.6	The most commonly defined incremental learning scenarios. . . . .	22
2.7	Visualization of inter-task confusion in class-incremental learning. . . . .	24
2.8	Schematic Overview of semantic segmentation with a FCN . . . . .	26
3.1	Categorization of continual learning algorithms by De Lange et al. [De 19] . . . . .	34
3.2	Schematic Overview of Learning without Forgetting . . . . .	39
4.1	Class- and domain-incremental learning scenarios for continual semantic segmentation. . . . .	45
4.2	Class distributions of Cityscapes and BDD100k . . . . .	46
4.3	Class distributions of the subsets of the disjoint Cityscapes setup . . . . .	47

4.4	Confusion matrices after training on the Cityscapes Disjoint Sequence. . . . .	53
4.5	Influence of the Memory Size on the mIoU (%) of different sample selection methods . . . . .	60
4.6	Influence of the Memory Size on the mIoU (%) of different sample selection methods . . . . .	60
5.1	Overview of methods to measure catastrophic forgetting. . . . .	64
5.2	Comparison of the original Dr. Frankenstein layer matching [Csi21] with the approach without an additional stitching layer . . . . .	67
5.3	Activation drift measured by Layer Stitching for domain- and class-incremental learning . . . . .	69
5.4	Activation drift measured by Layers Stitching (top) and Dr. Frankenstein (bottom) . . . . .	71
5.5	Activation drift measured by CKA for domain- and class-incremental learning . . . . .	74
5.6	The distance of $f_1$ and $f_0$ measured by $\ell_2$ -distance of the weights of the convolutional layer in the class-incremental setting (top) and the running mean of the batch normalization layers in the domain-incremental setting. . . . .	81
6.1	Bias values of the final convolutional layer after training on the <i>disjoint</i> PascalVoc-15-5 split. . . . .	91
6.2	Confusion matrices after training on PascalVoc-15-5 <i>disjoint</i> . . . . .	92
6.3	Activation drift between $f_1$ to $f_0$ for the <i>disjoint</i> and <i>full disjoint</i> Pascal-15-5 splits. . . . .	93
6.4	Visualizations of the segmentation maps for LwF, MAS and the resulting networks of MAS $f_{1,0}^{18}$ and $f_{1,0}^{19}$ . . . . .	95
6.5	Confusion matrices before (a) and after (b), (c) retraining the decoder on all classes of PascalVoc2012. . . . .	97
6.6	Activation drift for DeepLabV3+, U-Net, BiSeNet V2 and HRNetV2-W48 on Pascal-15-5. . . . .	101

6.7	The four different domain-incremental setups all starting from a model trained on Cityscapes and then are either adapted to <i>Rain</i> , <i>Night</i> , <i>Fog</i> or <i>Snow</i> subsets. . . . .	106
6.8	Comparison of the activation drift that models with and w/o improved training scheme are subjected to in domain-incremental learning. . . . .	108
6.9	Comparison of the Fourier amplitude spectra of Cityscapes, augmented CS and the ACDC subsets. . . . .	110
6.10	The influence of pre-training on the activation drift between $f_1$ to $f_0$ measured with layer stitching. . . . .	113
6.11	The influence of augmentations on the activation drift between $f_1$ to $f_0$ measured with layer stitching. . . . .	115
6.12	Change in mIoU on the first task after re-estimation of the population statistics of specific BN layers (horizontal axis). Re-estimation mostly affects the first BN layer and the last BN layers in each block's first layer. . . . .	118
6.13	Comparison of the activation drift between models trained with batch normalization and continual normalization. . . . .	119
6.14	Activation drift in domain-incremental learning with partially frozen layers. . . . .	121
6.15	Comparison of the activation drift between models trained on <i>Night</i> and <i>Fog</i> with sequentially adding pre-training with DINO, AutoAlb. and CN. . . . .	122
6.16	Comparison of the activation drift a model is subjected to when it is simply fine-tuned in (a) class-incremental and (b) domain-incremental learning. . . . .	130
7.1	Experimental setup and architecture choices to measure the effect of architecture on catastrophic forgetting . . . . .	136
7.2	Activation drift in domain-incremental learning for different encoders and decoders. . . . .	141

7.3	Confusion matrices after training on PascalVoc-15-5. The confusion matrix for ResNet-50 shows a severe bias to the background class and classes of the recent task (16-20). Using Swin-T or ConvNeXt as backbone with the same decoder head decreases the bias for the new classes and the background class. . . . .	144
7.4	Activation drift in domain-incremental learning for different encoders and decoders. . . . .	145
7.5	Forgetting and learning accuracy for ResNet → ConvNeXt individual vs. incremental changes . . . . .	149
A.1	Confusion matrices after training on PascalVoc-15-5 <i>overlapped</i> . . . . .	199

# List of Tables

4.1	Comparison of DeepLabV3+ (31M parameters) and ERFNet (1.2M parameters) in domain-incremental learning . . . . .	48
4.2	Comparison of continual learning methods on ERFNet by mIoU for the class-incremental setting. . . . .	51
4.3	Comparison of Continual Methods on ERFNet by mIoU for the domain-incremental setting. . . . .	55
4.4	Results in mIoU (%) of sample selection methods for domain-incremental setting with buffer size = 64. . . . .	58
4.5	Results in mIoU (%) of sample selection methods for class-incremental setting with buffer size = 64. . . . .	59
5.1	Comparison of EWC, replay and fine-tuning in the continual semantic segmentation. . . . .	66
5.2	Performance in mIoU (%) after after re-estimating the population statistics. . . . .	77
5.3	Weight distance calculated as $\ell_2$ -distance of the models parameters $\theta_0$ to $\theta_1$ . . . . .	80
5.4	Comparison of methods to measure catastrophic forgetting. . . . .	83
6.1	Results of semantic segmentation on Pascal-VOC 2012 in mIoU (%) on the <i>overlapped</i> , <i>disjoint</i> and <i>full disjoint</i> settings. . . . .	89
6.2	Decoder retraining results on Pascal-15-5. . . . .	96
6.3	Results on Pascal-15-5 in mIoU (%) when utilizing unbiased cross-entropy and weight normalization. . . . .	99

6.4	Results in mIoU (%) of fine-tuning for the <i>overlapped</i> , <i>disjoint</i> and <i>full disjoint</i> settings using fine-tuning with cross-entropy (CE) and unbiased cross-entropy (UNCE) . . . . .	102
6.5	Classification results on PascalVoc-15-5 in (%). . . . .	103
6.6	Results on $CS \rightarrow ACDC$ in mIoU (%) for each subset of ACDC. . . . .	107
6.7	The mean and standard deviations for the HSV channels of each subset. . . . .	110
6.8	Results on $CS \rightarrow ACDC$ in mIoU (%) for each subset of ACDC sing different pre-training and augmentation strategies (Augment.) . . . . .	112
6.9	Performance in mIoU (%) on $CS$ of the model $f_1$ after re-estimating all BN layer population statistics. . . . .	116
6.10	Results on $CS \rightarrow ACDC$ in mIoU (%) with batch- and continual normalization. . . . .	119
6.11	Performance and forgetting in mIoU (%) on $CS$ and the ACDC subsets <i>Night</i> and <i>Rain</i> . . . . .	120
6.12	Forgetting and Learning accuracy on $CS \rightarrow ACDC$ with incremental additions that increase the feature reuse, significantly reduces forgetting. . . . .	122
6.13	Forgetting and learning accuracy on <i>Night</i> and <i>Rain</i> with EWC and replay. . . . .	123
6.14	Forgetting and learning accuracy of different SegFormer-B2, DeepLabV3+ and ERFNet trained on $CS \rightarrow ACDC$ in mIoU (%) for each subset of ACDC. . . . .	124
6.15	Forgetting and learning accuracy in mIoU (%) of SegFormer-B2 with different augmentations. . . . .	124
6.16	Results of ERFNet [Rom18] on $CS \rightarrow ACDC$ in mIoU (%). . . . .	126
6.17	Results of BiSeNet V2 [Yu21] on $CS \rightarrow ACDC$ in mIoU (%). . . . .	126
6.18	Results of HRNetV2-W48 [Wan19] on $CS \rightarrow ACDC$ in mIoU (%) . . . . .	127
6.19	Results for $CS \rightarrow Rain \rightarrow Night$ with DeepLabV3+. . . . .	128

7.1	Overview of the selected encoder architectures with their respective top-1 ImageNet accuracy in (%) . . . . .	138
7.2	Overview of the selected decoder architectures with the number of parameters. . . . .	138
7.3	Comparison of the performance of different backbone architectures in domain-incremental learning. . . . .	140
7.4	Comparison of the performance of different decoder architectures in domain-incremental learning. . . . .	142
7.5	Comparison of the performance of different decoder architectures in domain-incremental learning. . . . .	143
7.6	Comparison of the performance of different backbone architectures in class-incremental learning. . . . .	143
7.7	Comparison of the performance of different normalization layers in class-incremental learning. . . . .	146
7.8	Results in mIoU (%) for the class-incremental setting with different decoder heads. . . . .	147
7.9	Architectural changes in the ablation study ResNet-50 → ConvNeXt inspired by Liu et al. [Liu22]. . . . .	148
B.1	Hyperparameters for the experiments of Sec. 6.1. . . . .	201
C.1	Hyperparameters for the experiments of Sec. 6.2. . . . .	203
C.2	List of augmentations used to increase feature reuse with there specified arguments and classes of Alumentations [Bus20]. . . . .	204
D.1	Hyperparameters for the experiments of Ch. 7. . . . .	205



# Acronyms

<b>ANN</b>	artificial neural network
<b>BN</b>	batch normalization
<b>CDA</b>	continual domain adaption
<b>CE</b>	cross entropy loss
<b>CiSS</b>	class-incremental semantic segmentation
<b>CKA</b>	centered kernel alignment
<b>CN</b>	continual normalization
<b>CNN</b>	convolutional neural network
<b>CS</b>	Cityscapes
<b>DiSS</b>	domain-incremental semantic segmentation
<b>DNN</b>	deep neural network
<b>EWC</b>	Elastic Weight Consolidation
<b>FE</b>	feature extraction
<b>FT</b>	fine-tuning

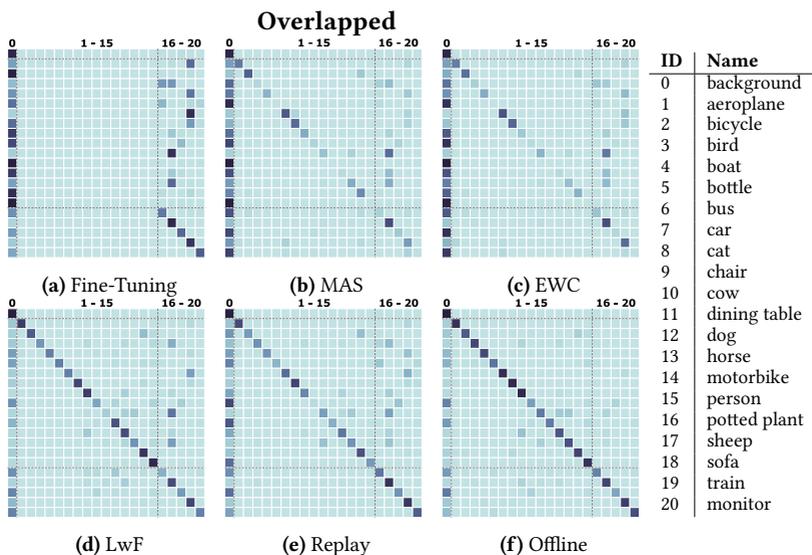
<b>GAN</b>	Generative Adversarial Network
<b>GN</b>	group normalization
<b>GSS</b>	gradient-based sample selection
<b>IN</b>	instance normalization
<b>LN</b>	layer normalization
<b>LwF</b>	Learning without Forgetting
<b>MAS</b>	Memory Aware Synapses
<b>mIoU</b>	mean intersection over union
<b>RSS</b>	representation-based sample selection
<b>SGD</b>	stochastic gradient descent
<b>SSL</b>	self-supervised learning
<b>UNCE</b>	unbiased cross entropy loss
<b>VT</b>	vision transformer





# A Confusion Matrices for the Pascal-15-5 Overlapped Setting

The confusion matrices in Figs. A.1a to A.1f for models trained in the *overlapped* setting show similar biases towards the background class and the most recent classes than models trained in the *disjoint* setting.



**Figure A.1:** Confusion matrices after training on PascalVoc-15-5 *overlapped*. The confusion matrix for a) Fine-Tuning shows a severe bias to the background class and the classes of the most recent task (16-20). EWC [Kir15] and MAS [Alj18] decrease the bias in exchange for worse accuracy on the most recent classes. Replay and LwF [Li18] reduce the bias towards new classes and the background.



## B Reproducibility of Sec. 6.1

### B.1 Training Protocol

The experiments in Sec. 6.1 follow the same evaluation protocol with the same train, test and validation splits as proposed by Cermelli et al. [Cer20]. The weights for ERFNet start from the same random initialization in all experiments, instead of using ImageNet pre-trained weights. The hyperparameters for the experiments are listed in Tab. B.1. The implementations for the augmentations are taken from Albumentations [Bus20].

**Table B.1:** Hyperparameters for the experiments of Sec. 6.1. Parameters in the middle column are for training on the first task, parameters in the right column for second task. Parameters in gray in the right column are the same as for the first task.

Parameter	Pascal-15-5 (Task 1)	Pascal-15-5 (Task 2)
Optimizer	SGD(weight_decay=0.0003, momentum: 0.9)	SGD(weight_decay=0.0003, momentum: 0.9)
Learning rate	0.07	0.0005
Learning rate scheduler	ReduceLROnPlateau(patience=8, factor=0.5)	ReduceLROnPlateau(patience=8, factor=0.5)
Batch size	16	16
Epochs	100	100
Training Augmentations	HorizontalFlip(p=0.5) RandomSizedCrop(h=512, w=512) Normalize()	HorizontalFlip(p=0.5) RandomSizedCrop(h=512, w=512) Normalize()

### B.2 Continual Hyperparameter Selection

The hyperparameters for the continual learning methods are selected according to the Continual Hyperparameter Framework of [De 19]. Therefore, in a

first step the learning rate is selected so that the model achieves the highest accuracy on the new task. In the second step the specific continual learning hyperparameters are tuned. This lead to the following hyperparameters for the respective methods:

- EWC<sup>1</sup>:  $\lambda = 10000$
- MAS:  $\lambda = 5000$
- LwF:  $\lambda = 6, \tau = 2$
- MiB:  $\lambda = 25, \alpha = 1$

In the experiments with UNCE and Weight Normalization the exact same hyperparameters are used.

---

<sup>1</sup> In order to use such a high value for  $\lambda$  we clip the gradient norm at a value of 10 to avoid exploding gradients.

## C Reproducibility of Sec. 6.2

### C.1 Training Protocol

The hyperparameters for the experiments are listed in Tab. C.1. The learning rate for the experiments in this section is tuned by running experiments with learning rates of  $LR \in \{0.1, 0.05, 0.01, 0.005, 0.001, 0.0005\}$ . Additionally, intermediate learning rates are tested between the best and second-best LR. For the pre-trained and augmentation models, the same LR is selected for optimization on Cityscapes. The parameters for fine-tuning and EWC are determined using the Continual Hyperparameter Framework [De 19].

**Table C.1:** Hyperparameters for the experiments of Sec. 6.2. Parameters in the middle column are for training on the Cityscapes, parameters in the right column for training on the ACDC subsets. Parameters in gray in the right column are the same as for the first task.

Parameter	Cityscapes	ACDC (subset)
Optimizer	SGD(weight_decay=0.0003, momentum: 0.9)	SGD(weight_decay=0.0003, momentum: 0.9)
Learning rate	$LR \in \{0.1, 0.05, 0.01, 0.005, 0.001, 0.0005\}$	0.005
Learning rate scheduler	PolynomialLR(power=0.9)	PolynomialLR(power=0.9)
Batch size	8	8
Epochs	200	120
Training Augmentations	HorizontalFlip(p=0.5) RandomSizedCrop(h=512, w=1024) Normalize()	HorizontalFlip(p=0.5) RandomSizedCrop(h=512, w=1024) Normalize()

## C.2 Models and Weights

The models and experiments are all implemented in PyTorch in combination with Pytorch Lightning. The PyTorch implementation of ERFNet is based on the code provided by [Rom18], which can be found at: [github.com/Eromera/erfnet\\_pytorch](https://github.com/Eromera/erfnet_pytorch), the DeepLabV3+ [Che18] implementation is adapted from *Segmentation Models PyTorch* [Iak19] and the SegFormer-B2 [Xie21] implementation from *Hugging Face Transformers* [Wol20]. The weights for the pre-trained ResNet-50 [He16] backbones are taken from:

- DINO [Car21]: [github.com/facebookresearch/dino](https://github.com/facebookresearch/dino)
- MoCo v3 [Che21a]: [github.com/facebookresearch/moco-v3](https://github.com/facebookresearch/moco-v3)
- BarlowTwins [Zbo21]: [github.com/facebookresearch/barlowtwins](https://github.com/facebookresearch/barlowtwins)
- SwAV [Car20]: [github.com/facebookresearch/swav](https://github.com/facebookresearch/swav)

The weights of ERFNet pre-trained with DINO and MoCo v3 can be found on [github.com/tobiaskalb/feature-reuse-css](https://github.com/tobiaskalb/feature-reuse-css).

## C.3 Augmentations

The augmentation schemes and their specific configurations that were used in the experiments are shown in Tab. C.2. The resulting augmentation scheme of AutoAlbum can be found at: [github.com/tobiaskalb/feature-reuse-css](https://github.com/tobiaskalb/feature-reuse-css).

**Table C.2:** List of augmentations used to increase feature reuse with there specified arguments and classes of Albumentations [Bus20].

Method	Albumentations Parameters
Distortion	ColorJitter(brightness=0.2, contrast=0.5, saturation=0.5, hue=0.2) ChannelShuffle(p=0.5)
Gaussian Blur	GaussianBlur(blur_limit=(3, 5))
Gaussian Noise	GaussNoise(var_limit=(30, 60))
AutoAlbument	Augmentation Config from <a href="https://github.com/tobiaskalb/feature-reuse-css">github.com/tobiaskalb/feature-reuse-css</a>

## D Reproducibility of Ch. 7

### D.1 Training Protocol

The hyperparameters for the experiments of Ch. 7 are listed in Tab. D.1.

**Table D.1:** Hyperparameters for the experiments of Ch. 7. Parameters in the middle column are for training on the domain-incremental setting, parameters in the right column for training on the class-incremental setting. Parameters in gray in the right column are the same as for the first task.

Parameter	Domain-Incremental	Class-Incremental
Optimizer	AdamW	AdamW
Learning rate	0.0001	0.0001
Learning rate scheduler	PolynomialLR(power=0.9)	PolynomialLR(power=0.9)
Batch size	6	6
Epochs	120	40
Training Augmentations	HorizontalFlip(p=0.5)	HorizontalFlip(p=0.5)
	RandomSizedCrop(h=769, w=769)	RandomSizedCrop(h=512, w=512)
	Distortion() Normalize()	Distortion() Normalize()

### D.2 Models and Weights

The experiments of Ch. 7 are conducted within the PyTorch-based *MMSegmentation* Framework [Con20]. Therefore, the implementations of the models and the ImageNet weights are directly taken from *MMSegmentation*. Only the implementation of NAT-T is directly adapted from [Has23].

